Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Žiga Simončič

Obogatitev interaktivnih eksperimentov z uporabo naprave Google Tango

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: doc. dr. Matija Marolt

Ljubljana, 2017

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Obogatitev interaktivnih eksperimentov z uporabo naprave Google Tango

Tematika naloge:

V okviru diplomskega dela preizkusite zmožnosti tablice Google Tango za obogatitev interaktivnih eksperimentov. Implementirajte rešitev, ki bo z vizualizacijo tokovnic in magnetnih polj obogatila električne eksperimente, ter hkrati zagotavljala tudi nadzor nad eksperimenti. Rešitev tudi ovrednotite.

Zahvaljujem se mentorju doc. dr. Matiji Maroltu za mentorstvo in usmeritev pri izdelavi diplomskega dela. Zahvaljujem se asistentu dr. Cirilu Bohaku za splošno pomoč in nasvete pri izdelavi diplomskega dela. Zahvaljujem se asistentu Matevžu Pesku za pomoč in uporabo električnih komponent. Posebna zahvala gre tudi moji družini in prijateljem, ki so me tekom študija vseskozi podpirali ali pa me kako drugače spodbujali. Dodatna zahvala gre zdaj že diplomiranemu prijatelju Nejcu Smrkolju Koželju za vse strokovne nasvete in modre misli.

Kazalo

Slike

Seznam uporabljenih kratic

kratica	angleško	slovensko
VR	virtual reality	navidezna resničnost
AR	augmented reality	obogatena resničnost
AV	augmented virtuality	obogatena virtualnost
ADF	area description file	datoteka opisa prostora
IDE	integrated development environment	integrirano razvijalsko okolje
GPIO	general purpose input/output	splošnonamenski vhodi/izhodi
XML	eXtensible markup language	razširljiv označevalni jezik
SLAM	simultaneous localization and mapping	sočasno sledenje in pomnjenje okolice

Povzetek

Naslov: Obogatitev interaktivnih eksperimentov z uporabo naprave Google Tango

Avtor: Žiga Simončič

V okviru te diplomske naloge smo testirali razvojno tablico Google Tango in z njo implementirali prototip za vizualizacijo električnih eksperimentov (tokovi, magnetna polja ...). Kot glavna razvojna orodja smo uporabili Unity3D in Visual Studio s programskim jezikom C#. Vizualizacijo tokov in magnetnih polj smo dosegli z risanjem puščic, ki so definirane z Bezierovimi krivuljami. Za večjo interaktivnost smo uporabili še Raspberry Pi, ki smo ga povezali s tablico Tango in eksperimentom, kar nam je omogočilo oddaljeno upravljanje eksperimenta. Ustrezno poravnavo vizualizacije smo dosegli z uporabo pomnjenja prostorov, ki ga omogoča platforma Google Tango. Med pomnjenjem prostora lahko na posamezne dela eksperimenta postavljamo oznake, s katerimi si olajšamo izdelavo vizualizacije. Implementirali smo tudi vtičnik za Unity3D, ki te oznake prenese s tablice na računalnik. Prototip smo ovrednotili in podali napotke za izdelavo, kjer smo ločili dva tipa eksperimentov - mobilne, ki se pogosto prestavljajo v različne prostore, in statične, ki so dlje časa v istem prostoru. Pri vizualizaciji električnih eksperimentov že nekaj milimetrsko odstopanje vizualizacije povzroči opazen zamik, kar pomeni, da moramo biti pri izdelavi v Unity3D izjemno natančni.

Ključne besede: navidezna resničnost, obogatena resničnost, project tango, google tango, vizualizacija, električni eksperiment.

Abstract

Title: Augmentation of interactive experiments with a Google Tango device

Author: Žiga Simončič

In this thesis, we tested the Google Tango development tablet and built a prototype for visualization of electric experiments (currents, magnetic fields, etc.). As the main development tools, we used Unity3D and Visual Studio with C# programming language. We achieved current and magnetic field visualization by drawing arrows, which are defined by Bezier curves. For a better interactivity, we also used the Raspberry Pi, which we connected to the Tango tablet and the experiment. This enabled us to manage the experiment remotely. By utilizing Google Tango's Area Learning feature, we were able to properly align the visualization. While learning an area, we can place markers on different parts of the experiment, which helps us build the visualization. We implemented a plugin for Unity3D, which transfers those markers from the tablet to the computer. We evaluated the prototype and presented instructions on how to build similar systems. We considered two possibilities: mobile experiments, which will frequently move between different places, and static experiments, which will stay in the same place for a longer time. When visualizing electric experiments, even a few millimetre deviation of visualization can be quite noticeable, which means we need to be very precise when creating the visualization in Unity3D.

Keywords: virtual reality, augmented reality, project tango, google tango, visualization, electric experiment.

Poglavje 1

Uvod

Dandanes sta pojma navidezna resničnost (*virtual reality* - VR) in obogatena resničnost (*augmented reality* - AR) zelo priljubljena. Vprašanje pa ostaja, kaj sploh sta oziroma, kako sta ta dva pojma povezana. Opozorili bi, da za te pojme ni stroge definicije [4]. Obstaja tudi še nekaj drugih izrazov (npr. *mixed reality*), kar še povečuje zmedo. Opisali bomo, kaj se dandanes največkrat razume pod temi pojmi.

Pojem navidezna resničnost se nanaša na navidezno okolje, v katerem ima uporabnik občutek fizične prisotnosti. Uporabnik ima možnost interakcije s tem okoljem, simulirane pa so lahko tudi zakonitosti resničnega sveta [21]. Uporabniku je pogled na resnični svet povsem zastrt. Naprave za VR se po navadi pojavljajo v obliki očal oziroma čelad. Pomembnejša predstavnika VR-naprav sta Oculus Rift in HTC Vive.

Obogatena resničnost uporabniku ne zastre pogleda in ne simulira novega, sintetičnega sveta, ampak mu (o)bogati obstoječi (resnični) svet in okolje. V naš pogled na svet doda grafične elemente, ki prikažejo dodatne informacije, nadomestijo obstoječe objekte ali se kako drugače vključijo v svet. Tovrstna tehnologija pridobiva priljubljenost v raznih ustanovah (npr. muzejih [11]) za izboljšanje uporabniške izkušnje. Primer AR-naprav je npr. Google Tango, do neke mere tudi Google Glass.

Razvoj teh tehnologij sega že okoli 50 let v preteklost. Za prvi VR-sistem

se smatra naglavni prikazovalnik "The Sword of Damocles",[1] ki ga je leta 1968 realiziral Ivan Sutherland [19]. V splošnem so bile naprave v preteklosti večinoma zelo specializirane, uporabljale pa so se npr. za vizualizacijo v medicini ali simulacijah za učenje. Zaradi splošnega razvoja tehnologije so danes tovrstne naprave za uporabo enostavnejše. Prav tako lahko na njih poganjamo kompleksnejše aplikacije, kot je bilo to možno v preteklosti. V zadnjih nekaj letih so postale tudi veliko bolj cenovno dostopne in znane.

Današnje naprave se promovirajo z različnimi demo aplikacijami v obliki iger, vendar jih lahko uporabimo tudi v druge namene. Primer uporabe je obogatitev uporabniške izkušnje v različnih ustanovah. Google s svojo tablico s podporo za Google Tango (v nadaljevanju tudi: tablica Tango) že sodeluje z različnimi muzeji in tako popestri izkušnjo obiskovalcev ter jih seznanja s to tehnologijo [11]. Obiskovalci lahko s tablico Tango opazujejo razstavne eksponate. Glede na lokacijo in orientacijo tablice se na njenem zaslonu prikazujejo in dorisujejo dodatne informacije na obstoječi svet, ki ga vidi kamera. Obiskovalci lahko s tablico "vidijo" okostnjaka v sarkofagu, barvo na starih reliefih, vizualizacijo stare in manjkajoče arhitekture ipd.

V sklopu tega diplomskega dela smo se tudi mi osredotočili na tehnologijo Google Tango. Želeli smo preizkusiti, kako lahko s tehnologijo mobilne obogatene resničnosti, ki jo omogočajo naprave, združljive z Google Tango, obogatimo interaktivne eksperimente, kot jih lahko najdemo v muzejih in podobnih ustanovah. Zadali smo si cilj raziskati in testirati delovanje tablice Tango na konkretnem področju - električnih eksperimentih. Eksperiment smo želeli obogatiti z grafičnimi elementi, ki prikazujejo in vizualizirajo dodatne informacije o magnetnih poljih, tokovih, elementih itd. Za boljšo izkušnjo smo hoteli uporabiti Raspberry Pi in z gumbom upravljati eksperiment. Z uporabo povezave Bluetooth smo želeli Raspberry Pi še povezati s tablico Tango in sinhronizirati vizualizacijo z dejanskim stanjem eksperimenta.

Raziskovali smo po metodah pilotne študije (angl. *Pilot case*) in študije

[1] Več: `https://en.wikipedia.org/wiki/The_Sword_of_Damocles_(virtual_reality)`

izvedljivosti (angl. *Feasibility study*).

Prvo metodo smo uporabili, ker smo v sklopu diplomskega dela izdelovali prototip, s katerim smo želeli pokazati uporabo Google Tango na konkretnem primeru električnih eksperimentov. Druga metoda je v tem smislu zelo povezana s prvo. Treba je bilo raziskati oziroma poskusiti, če je prototip, ki smo ga imeli namen realizirati, sploh izvedljiv. V primeru uspešne realizacije nas je zanimalo še, kako dobro se je obnesel v praksi.

V 2. poglavju predstavimo področje, kjer bolj podrobno opišemo prej omenjene pojme in naprave. Glavna orodja in tehnologije, ki smo jih uporabili za doseg zastavljenih ciljev, so opisani v poglavju 3, testiranje tablice Tango pa je opisano v poglavju 4. V 5. poglavju opišemo prototip in razčlenimo njegovo implementacijo, v 6. poglavju pa predstavimo rezultate in sklep. Sledi še zaključek.

Poglavje 2

Pregled področja

V tem poglavju bolj podrobno predstavimo področja navidezne in obogatene resničnosti, zraven pa se dotaknemo še področja mešane resničnosti. Zatem predstavimo še bolj znane tehnologije in naprave z vseh treh področij. Ker gre za relativno novo potrošniško tehnologijo, teh naprav še ne vidimo v širši uporabi. Pomembna dejavnika sta tudi cena in namen. Za številne je strošek previsok, veliko ljudi pa teh tehnologij sploh ne pozna, saj se najbolj agresivno oglašujejo v zabavni industriji. Eden večjih predstavnikov te industrije so računalniške igre, kar pa ne pomeni, da se teh naprav ne da izkoristi tudi v druge namene.

2.1 Navidezna resničnost

Začetek navidezne resničnosti ni povsem definiran. Prvo idejo navidezne resničnosti, kot jo poznamo danes, je predstavil Ivan Sutherland leta 1965 [19]. Na osnovi svoje ideje je kasneje izdelal napravo, ki se dojema za prvi naglavni prikazovalnik (angl. *Head Mounted Display - HMD*),[1] ki ga je imenoval "The Sword of Damocles".[2] Prikazovalnik je omogočal pogled na svet, ki se je prilagajal položaju in orientaciji glave. Skozi čas so znanstveniki in

[1] Danes bi temu rekli očala ali čelada.

[2] Več: `https://en.wikipedia.org/wiki/The_Sword_of_Damocles_(virtual_reality)`

podjetja te prikazovalnike izpopolnjevali ter uporabljali za različne namene (npr. vizualizacija podatkov, obhodi znamenitosti, simulacije za učenje) [19].

Podobni sistemi in koncepti so sicer obstajali že pred idejo Ivana Sutherlanda (npr. Sensorama[3]). V nasprotju s takrat obstoječimi sistemi pa je Ivan Sutherland poudarjal pomembnost interaktivnosti, kar je vodilo v razvoj omenjenega prvega naglavnega prikazovalnika [4].

Za virtualno resničnost obstaja več definicij [4]. Zeltzerjeva definicija pa povzame vse bistvene lastnosti: navidezna resničnost je interaktivna izkušnja z občutkom prisotnosti v simuliranem, avtonomnem svetu [33].

Naprave za navidezno resničnost uporabniku popolnoma zastrejo pogled, kar stimulira občutek pripadnosti oziroma prisotnosti v navideznem svetu. Opremljene so s senzorji za zaznavanje premikanja glave, v nekaterih primerih tudi hoje. Za še večjo interaktivnost so na voljo tudi brezžični krmilniki.

Priljubljeni napravi za navidezno resničnost sta Oculus Rift in HTC Vive.

2.2 Obogatena resničnost

Obogatena resničnost temelji na združevanju digitalnih (umetnih) informacij z resničnim svetom [4]. V naš pogled na svet doda grafične elemente, ki nadomestijo ali dopolnjujejo obstoječe objekte, prikazujejo dodatno informacijo oziroma se na kak drug način vključijo v svet.

Tako kot pri navidezni resničnosti je tudi tukaj težko dokončno postaviti začetek in definicijo obogatene resničnosti. Nekateri obogateno resničnost dojemajo kot podmnožico navidezne resničnosti, nekateri pa ravno obratno [4].

Obogatena resničnost se pojavlja v dveh oblikah: naglavnih prikazovalnikih (očala/čelade, ki ne zastrejo pogleda) in tablicah oziroma telefonih. Poznamo še eno vejo obogatene resničnosti, ki se imenuje okoljska obogatena resničnost (angl. *Spatial Augmented Reality - SAR*) [4]. Ideja te resničnosti je integracija prikaza obogatene resničnosti neposredno v uporabnikovo okolje.

[3]Sensorama: `https://en.wikipedia.org/wiki/Sensorama`

Največkrat se za to uporabljajo projektorji ali vgrajeni zasloni. Uporabniku v takem primeru ni treba nositi nobene vrste naglavnih zaslonov, kar je tudi glavna prednost te resničnosti [25].

Bolj znana predstavnika obogatene resničnosti sta napravi Google Glass (očala) in Google Tango (tablica, telefon).

2.3 Mešana resničnost

Leta 1994 sta profesor Paul Milgram in raziskovalec Fumio Kishino v članku [21] razčlenila definicijo mešane resničnosti. Naprave mešane resničnosti sta razvrstila v šest razredov. Ugotovila sta, da se obogatena resničnost v literaturi večinoma nanaša samo na enega od šestih razredov. Kot nasprotje obogateni resničnosti sta definirala še pojem obogatene virtualnosti (angl. *Augmented Virtuality - AV*), katere značilnost je bogatitev navideznega sveta z objekti iz resničnega sveta. Vseh šest razredov skupaj zajema vse vrste naprav, ki uporabniku ne zastrejo pogleda. Mešano resničnost sta torej definirala kot nadpomenko obogateni resničnosti in obogateni virtualnosti, ki zajema vse primere od popolnoma resničnega do popolnoma navideznega sveta.

Priljubljen predstavnik tehnologije mešane resničnosti so očala Microsoft HoloLens.

2.4 Naprave

2.4.1 Microsoft HoloLens

Microsoft HoloLens [20] spada v t. i. skupino *mešane resničnosti*. Združuje obogateno resničnost in obogateno virtualnost.

Fizično so to očala, ki nam na obstoječe okolje dorisujejo različne elemente in objekte. Pri Microsoftu dorisane elemente imenujejo holograme (zato tudi ime HoloLens).

Za interakcije s hologrami so na voljo trije glavni načini: pogled, kretnje in glas.

Očala imajo naslednje senzorje: pospeškomer, žiroskop, magnetomer, globinsko kamero, HD-video kamero, štiri kamere za razumevanje okolja, štiri mikrofone in svetlobni senzor. Poganja jih po meri izdelan procesor "Microsoft Holographic Processing Unit" (HPU), ki je osnovan na 32-bitni Intel arhitekturi. Imajo 2 gigabajta delovnega pomnilnika, 64 gigabajtov pomnilnika Flash in vgrajene zvočnike. Na njih teče operacijski sistem Windows 10.

Cena razvijalske različice očal ob času pisanja znaša 3.000 $, izdana pa so bila 30. marca 2016.

2.4.2 Oculus Rift

Facebook je pred kratkim kupil podjetje Oculus in tako pridobil njihovo napravo za VR, Oculus Rift [24].

V nasprotju z Microsoft Hololens očala Oculus Rift uporabniku v celoti zastrejo pogled. Resnični svet in okolje nadomestijo z virtualnim, ki se ustrezno prikazuje na zaslonu znotraj očal pred uporabnikovimi očmi.

Očala znajo zaznavati hojo in se prilagajati gibom glave. Interakcija z navideznim svetom pa ni tako brezhibna, saj za to potrebujemo posebne krmilnike, imenovane *Touch controllers*.

Očala podpirajo resolucijo 2160 x 1200 pikslov. Niso samostojna, saj morajo biti ves čas uporabe povezana z računalnikom (žično). Na spletni strani imajo objavljene minimalne in priporočene sistemske zahteve[4] za zagon aplikacij. Razvijamo lahko aplikacije za računalnike, mobilne naprave in splet. Na voljo imamo tudi programska paketa za razvoj s 3D-pogonoma Unity in Unreal Engine 4.

Oculus Rift je bil izdan 28. marca 2016, cena kompleta očal in dveh krmilnikov pa je ob času pisanja 598 $.

[4]Oculus Rift sistemske zahteve: `https://support.oculus.com/1633938460220125/`

2.4.3 HTC Vive

HTC Vive [15] je fizično podoben Oculus Riftu - to so očala, ki uporabniku zastrejo pogled, na voljo pa imamo dva krmilnika, s katerima interaktiramo s svetom.

Očala delujejo na resoluciji 2160 x 1200 pikslov. Vsebujejo pospeškomer, žiroskop, magnetomer, sprednjo kamero in laserski sledilni sistem "Lighthouse",[5] ki omogoča natančno zaznavanje sprehajanja po prostoru in robustno sledenje krmilnikov. HTC Vive ima dobro integracijo s platformo SteamVR, podpira pa tudi razvoj s 3D-pogonoma Unity in Unreal Engine 4.

Izdan je bil 5. aprila 2016, njegova cena ob času pisnja pa je 899 €.

2.4.4 Google Tango

Strogo rečeno, je Tango [27] le programska platforma za uporabo in razvoj AR-aplikacij. Prvotno je bil projekt imenovan *Project Tango*, nato pa so ga preimenovali samo v *Tango*. Izdan je bil 5. junija 2014.

Platforma omogoča uporabo AR na mobilnih napravah (tablice, pametni telefoni itd.). Za uporabo vseh zmožnosti platforme pa naprave potrebujejo posebno strojno opremo. Google zato sodeluje z različnimi podjetji (npr. Lenovo), da v izbrane linije mobilnih naprav vgradijo vso potrebno opremo za uporabo te platforme. Končni uporabnik po funkcionalnosti ne čuti bistvene razlike. Naprave še vedno tečejo na operacijskem sistemu Android in še vedno so na voljo vse funkcionalnosti, ki bi jih pričakovali od mobilnega telefona ali tablice. V trgovini Google Play so dodatno na voljo aplikacije, ki za delovanje potrebujejo platformo Tango. Tako želi Google rešiti težavo, kako na čim bolj eleganten način približati to tehnologijo širši publiki.

Naprava za podporo platforme Tango potrebuje širokokotno (angl. *fisheye*) kamero, globinsko kamero, natančno časovno označevanje senzorskih podatkov in programski paket, ki omogoča uporabo teh komponent [28].

Google Tango omogoča t. i. "SLAM" (angl. *Simultaneous localization and*

[5] Več o tehnologiji Lighthouse: `https://xinreality.com/wiki/Lighthouse`

mapping), kar pomeni, da omogoča sočasno pomnjenje in sledenje trenutne lokacije v prostoru.

Tudi Google je svojo platformo promoviral z izdelavo obogatene resničnosti za muzeje [11].

V trgovini Google Play najdemo kar nekaj aplikacij Tango. Večinoma so to eksperimentalne aplikacije ali igre, vmes pa najdemo tudi kakšno bolj dodelano in uporabno aplikacijo. Izpostavili bi aplikacijo za navidezno opremljanje prostora s pohištvom,[6] aplikacijo za merjenje dolžine[7] (napravo uporabimo namesto metra) in Googlove aplikacije za pomnjenje ter razpoznavanje prostorov.[8]

2.4.5 Aplikacije VR in AR

V članku [9] avtorji predstavijo uporabo HTC Vive v medicinske namene. Opisujejo izdelavo lastnega modula s pomočjo knjižnice OpenVR in integracijo očal s platformo MeVisLab. Rezultat sta enostavna uporaba in vizualizacija medicinskih podatkov (npr. 3D-vizualizacija organov, okostja).

Čeprav je Oculus Rift namenjen primarno igram, pa v članku [13] avtorji opisujejo uporabo v medicinske namene. Ugotovili so, da lahko pacienta zamotijo z virtualnim svetom in mu tako olajšajo bolečine.

Očala Microsoft HoloLens obljubljajo brezhibno združitev resničnega in navideznega sveta. Z njimi so avtorji v diplomski nalogi [29] raziskali uporabnost očal HoloLens za zaznavanje podmornic oziroma morskih min. Sodelovali so s švedskim podjetjem *SAAB Defence and Security*, ki izdeluje bojna letala, radarje in podobne izdelke za potrebe vojske. Za vizualizacijo so uporabili pogon Unity. Ugotovili so, da je z očali možno uspešno prikazati ustrezne hologrami, vendar se v praksi zaradi morskih valov in nestabilnosti

[6]Povezava do aplikacije: `https://play.google.com/store/apps/details?id=com.Elementals.ARHomeDesigner`

[7]Povezava do aplikacije: `https://play.google.com/store/apps/details?id=com.google.tango.measure`

[8]Povezava do aplikacije: `https://apkpure.com/tango-explorer-tool/com.projecttango.tangoexplorer`

morskih plovil zamikajo, ker očala izgubijo prostorsko predstavo. Omejitev je bila tudi procesorska moč.

V člankih [16] in [17] so avtorji opisali sistema za navigacijo po notranjih prostorih. V prvem predstavijo uporabo naprave Google Tango za zaznavanje ovir, kar bi bilo v pomoč ljudem, ki imajo težave z vidom. V drugem pa opisujejo uporabo za navigacijo notranjih prostorov.

Avtorji članka [23] so razvili sistem, ki združuje tehnologiji Google Tango in Google Glass. Sistem omogoča označevanje predmetov v prostoru. Oznake lahko ustvarimo preko glasovnega ukaza in nato izbire predmeta, ki naj se označi.

Omenili smo že, da se Google Tango uporablja tudi v muzejih. Inštitut za umetnost v Detroitu uporablja to tehnologijo za izboljšanje uporabniške izkušnje [11]. Obiskovalcem je za ogled na voljo tablica Lenovo Phab 2 Pro. Na njej je nameščena aplikacija, s katero lahko obiskovalci prosto hodijo okoli in si ogledujejo razstavne eksponate. Aplikacija glede na položaj in orientacijo tablice ustrezno prikazuje grafične elemente in dodatne informacije o eksponatih. Primeri[9] so vizualizacija okostnjaka ali mumije v sarkofagu, vizualizacija starih zgradb in arhitekture ter vizualizacija različnih grafičnih elementov, ki obogatijo stare risbe in objekte (v tem primeru obarvajo stare reliefe). Na voljo so tudi interaktivne dejavnosti v obliki kvizov in iger. Za lažjo navigacijo pa aplikacija obiskovalcem v svet postavlja oznake in začrta pot do želenega cilja (podobno kot GPS-naprave, le da se v tem primeru riše na tla in v prostor). Tehnologijo Google Tango so uporabili tudi v Muzeju umetnosti in znanosti v Singapurju [14]. S 3D-pogonom Unity3D so ustvarili vizualizacijo preko 1000 kvadratnih metrov velikega pragozda, skozi katerega se lahko sprehajamo in si ogledujemo okolje. Drevesa in druge ovire so ustrezno poravnali s stenami muzeja.

Nekateri muzeji pa dosegajo obogateno resničnost brez uporabe Google Tango in podobnih naprav [10]. Na sistemih Android in iOS ponujajo aplikacije, ki delujejo na navadnih mobilnih napravah. Ker navadne mobilne

[9]Več primerov: `https://www.guidigo.com/ar`

naprave ne vsebujejo senzorjev, kot jih vsebujejo naprave Tango, se morajo aplikacije zanašati na zunanje signale. To so lahko GPS, WiFi, Bluetooth ali kateri drugi signali, ki aplikaciji posredujejo informacijo o lokaciji (npr. Bluetooth oddajniki v vsaki sobi ali eksponatu). V nekaterih primerih lahko eksponat tudi slikamo. Glede na lokacijo ali s slike prepoznan razstavni eksponat pa se v aplikaciji prikažejo dodatne informacije, po navadi v obliki interaktivnega kataloga. S tem dobimo podoben rezultat kot pri tablicah Tango, vendar z manjšo natančnostjo in brez prilagajanja grafičnih elementov glede na gibanje in orientacijo.

Poglavje 3

Uporabljene tehnologije in orodja

Cilj diplomskega dela je bila vizualizacija električnih tokov in magnetnih polj na primeru električnih eksperimentov z uporabo tablice, ki podpira platformo Google Tango. Za večjo interaktivnost smo eksperiment želeli preko tablice tudi krmiliti. Vizualizacijo smo dosegli z risanjem animiranih puščic (npr. puščic, ki se premikajo po žici), večjo interaktivnost pa z uporabo Raspberry Pi in povezave Bluetooth.

V nadaljevanju podrobneje predstavimo uporabljene tehnologije in orodja.

3.1 Tablica Google Tango

Uporabljali smo enako tablico Google Tango, kot je na sliki 3.1. Uradni naziv konkretne tablice je *Project Tango Development Kit* (*Project Tango* zaradi nekoliko starejše izdaje). Drugo ime je tudi *Yellowstone*.

Slika 3.1: Uporabljena tablica Google Tango

Na njej je nameščen operacijski sistem Android, skupaj z vsemi aplikacijami, ki bi jih pričakovali (Google aplikacije, Tango aplikacije, telefon itd.). Jedro platforme Tango (aplikacija/storitev *Tango Core*) je imelo v času izdelave diplomskega dela tudi nekaj posodobitev.

Tablica ima IPS-zaslon z resolucijo 1920 x 1200 pikslov, ki je odporen na praske. Dimenzije so 119,77 x 196,33 x 15,36 milimetrov, teža pa okoli 370 gramov. Ima 128 gigabajtov notranjega spomina, ki ga je možno razširiti z uporabo kartic micro SD. Omogoča nam tudi vstavitev kartice nano SIM, kar pomeni, da lahko tablico uporabljamo kot normalno mobilno napravo oziroma kot telefon. Podpira najnovejše tehnologije za mobilni prenos podatkov (4G LTE), povezavo na brezžična internetna omrežja (WiFi), NFC (*Near Field Communication)* in povezavo Bluetooth. Na voljo imamo tudi naslednje priklope: Micro-USB, Micro HDMI, 3.5mm priklop za slušalke in priklop za polnilno postajo.

Vgrajen je štiri-jedrni procesor NVIDIA Tegra K1, ki deluje na taktu 2,3 GHz in vključuje tudi grafični procesor s 192 jedri CUDA (*Compute Unified Device Architecture*). Na voljo imamo 4 gigabajte notranjega pomnilnika. Za podporo platforme Tango pa ima naslednje naprave: kamero za zaznavanje premikov, projektor infrardečih žarkov za zaznavanje (3D) globine, pospeškometer, svetlobni senzor, barometer, kompas, GPS in žiroskop.

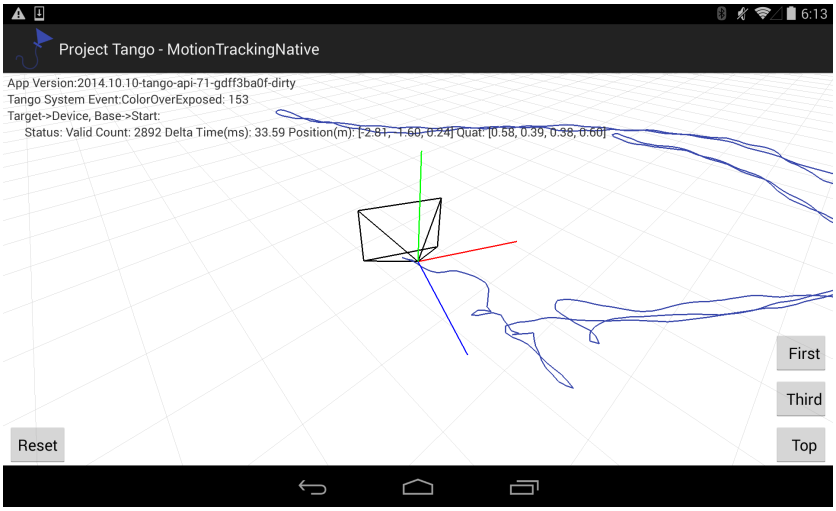
Tango je zasnovan na treh glavnih konceptih:

- sledenje gibanja (angl. *Motion Tracking*),
- pomnjenje prostorov (angl. *Area Learning*) in
- zaznavanje globine (angl. *Depth Perception*).

Sem bi lahko dodali še storitev vizualne postavitve (angl. *Visual Positioning Service*). To je sistem za mapiranje notranjih prostorov, vendar je trenutno v fazi zaprte bete.

3.1.1 Sledenje gibanja

Spremljanje gibanja omogoča napravi beleženje gibanja skozi prostor in omogoča razvijalcu dostop do podatkov o pozi (angl. *pose*). Poza je kombinacija položaja in orientacije (rotacije). Enota položaja je v metrih, rotacija pa v kvaternionih. Prikaz beleženja gibanja je viden na sliki 3.2.



Slika 3.2: Prikaz beleženja gibanja na napravi Google Tango [22]

Vrednost podatkov je relativna glede na *Frame of Reference*. To je fiksna poza v 3D-koordinatnem sistemu. Vse stvari se računajo relativno od te poze.

Primer take poze je začetna poza. Ko zaženemo aplikacijo, se zabeležita trenutni položaj in rotacija kot [0, 0, 0]. Vsi nadaljnji premiki in rotacije se računajo glede na to začetno vrednost.

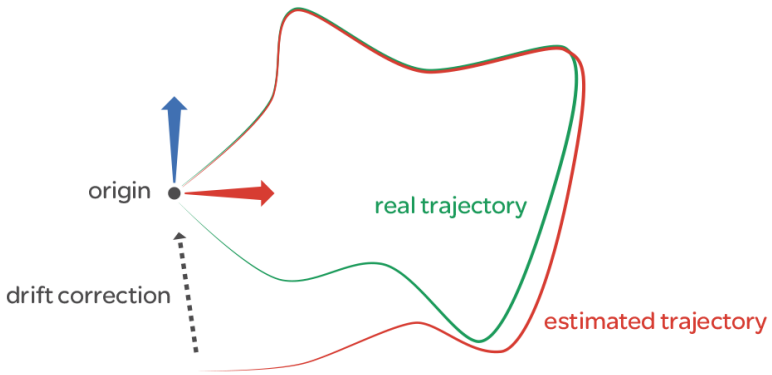
Za izračun poz Tango poleg žiroskopa in pospeškometrov uporablja tudi širokokotno kamero, s katero doda še vizualno informacijo.

Sledenje gibanja pa ima dve večji omejitvi. Ne zapomni si samega območja oziroma okolja in skozi čas postaja nenatančen. Slednje se zgodi zaradi majhnih napak, ki se počasi seštevajo v večjo napako. Obe omejitvi rešuje koncept pomnjenja prostorov.

3.1.2 Pomnjenje prostorov

Pomnjenje območij daje napravam možnost pomnjenja ključnih vizualnih informacij (lastnosti) o okolici. Informacije, ki si jih zapomni, lahko kasneje uporabi za razpoznavanje okolice. Da lahko to stori hitro, informacije pretvori v matematične opise, ki jih indeksira.

Ta koncept rešuje tudi težavo majhnih napak. Te se čez čas seštevajo in povzročijo večjo napako. Ker pa si Tango zapomni prostore in okolje, lahko glede na zapomnjeno pozicijo svojo pot z napakami ustrezno popravi. To se imenuje popravljanje drsenja (angl. *drift correction*). Delovanje je orisano na sliki 3.3.



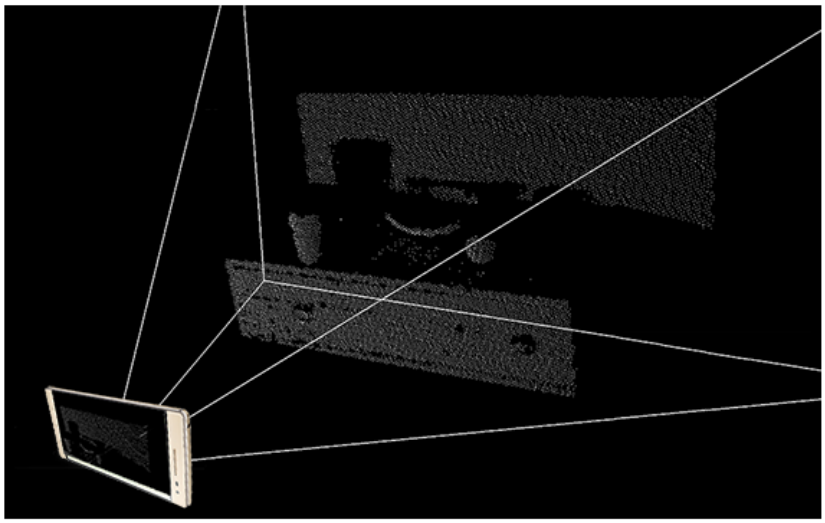
Slika 3.3: Oris delovanja popravljanja napak [2]

Podatke o prostorih in okolju si Tango hrani v datotekah za hranjenje opisa prostora (angl. *Area Description File - ADF*).

3.1.3 Zaznavanje globine

Omogoča nam zaznavanje globine in razdalj v prostoru. Za zaznavanje globine obstaja več načinov. Kateri način je uporabljen, je odvisno od proizvajalca mobilne naprave, na kateri teče Tango.

Do informacij o globini lahko dostopamo preko oblaka točk (angl. *pointcloud*). V realnem času Tango zajame globinske podatke in nato vrne množico [x, y, z] točk, ki predstavljajo položaje v 3D-prostoru. Zajem oblaka točk je prikazan na sliki 3.4.



Slika 3.4: Zajemanje oblaka točk [7]

3.2 3D-pogon Unity3D

Google Tango nam za razvoj ponuja tri možnosti: C, Javo in Unity3D [30].

Unity3D je 3D-pogon, primarno namenjen izdelavi iger, je pa dovolj splošen, da ga lahko uporabimo tudi za različne vizualizacije, simulacije in

druge vsebine. Z njim lahko razvijemo aplikacije za večino večjih platform: *Windows*, *MacOS*, *Linux*, *iOS*, *Android*, *Xbox*, *Playstation* in splet. Izdan je bil leta 2005, konstantno pa pridobiva priljubljenost zaradi dostopnosti in enostavnosti. Prednost je tudi vizualna povratna informacija, saj bi v primeru izbora jezika C ali Jave kodo morali pisati "na slepo". To pomeni ročno pisanje koordinat, poravnavanje in tudi izdelavo oziroma oblikovanje grafičnih elementov. Unity nam v tem pogledu omogoča bistveno hitrejši in bolj intuitiven razvoj. Z njim imamo tudi največ izkušenj. To so razlogi, da smo ga v sklopu tega diplomskega dela uporabili.

Razvoj si dodatno olajšamo z uporabo trgovine Unity Asset Store. Z enim klikom lahko prenesemo že dokončane modele, animacije, vtičnike in druge komponente v svoj projekt Unity. Na trgovino lahko objavlja kdor koli. Nekatere vsebine so brezplačne, bolj profesionalne pa so tudi plačljive. V vsakem primeru nam ob razvoju aplikacije oziroma igre lahko to zelo olajša delo.

Na voljo je več različic pogona Unity. Ena izmed različic je brezplačna, druge so plačljive. V osnovi so vse različice enake, z višjo ceno se nam omogočijo le boljše funkcionalnosti in storitve v zvezi z izdajo aplikacije. Za nas so zadostovale funkcionalnosti brezplačne različice.

Unity3D za razvoj omogoča uporabo treh jezikov: C# (različica Mono), UnityScript in Boo. Uporabili smo jezik C#, saj imamo z njim največ izkušenj in je v skupnosti Unity najbolj razširjen. Tudi Googlov paket Tango za Unity ima skripte napisane v tem jeziku. Unity uporablja C# na platformi Mono, ki ustreza standardu ECMA [8]. To zagotavlja kompatibilnost z več platformami, vendar smo, kar se tiče funkcionalnosti, v primerjavi s C# na platformi .NET nekoliko v zaostanku.

Kot urejevalnik kode (*IDE*) smo uporabljali Visual Studio, ki ima dobro podporo za razhroščevanje aplikacij Unity.

3.3 Visual Studio

Visual Studio je integrirano razvojno okolje (angl. *IDE - Integrated development environment*) podjetja Microsoft. Na voljo imamo več različic, konkretno smo uporabljali Visual Studio 2015 Community Edition. Tako imenovane *Community* različice so brezplačne, medtem ko sta različici *Professional* in *Enterprise* plačljivi, ponujata pa več funkcionalnosti, ki so predvsem osredotočene na projektno delo v skupini oziroma podjetju.

Urejevalnik ima podporo za zelo velik nabor različnih programskih jezikov in platform. Najbolj v ospredju pa je Microsoftova platforma .NET, v kateri lahko razvijamo z različnimi programskimi jeziki (npr. Visual C++ ali C#). Zelo dobro podprt je razvoj v jeziku C#.

Unity ima z razvojnim okoljem Visual Studio zelo dobro integracijo. Poleg drugih funkcionalnosti (npr. dopolnjevanje kode) se lahko pripnemo na proces Unity. To nam omogoča postavitev kontrolnih točk (angl. *breakpoints*) in uporabo razhroščevalnika Visual Studio. Ko je aplikacija začasno zaustavljena, lahko z njim preverimo trenutno stanje naših spremenljivk in popravljamo delčke kode.

3.4 Bezierove krivulje

Za vizualizacijo električnih tokov in magnetnih polj smo se odločili uporabiti Bezierove krivulje [3]. To so parametrične krivulje, ki se uporabljajo v računalniški grafiki in nam omogočajo neomejeno povečavo oziroma skaliranje. Z njimi lahko intuitivno predstavimo tudi kompleksnejše oblike krivulj.

Oblika krivulje je definirana s kontrolnimi točkami. Red oziroma stopnja določa, kakšna je lahko oblika krivulje. Število kontrolnih točk je $stopnja + 1$. Stopnja 1 pomeni linearno krivuljo, stopnja 2 kvadratno, stopnja 3 pa kubično. Po istem postopku se določajo tudi krivulje višjih stopenj.

Bezierove krivulje zapišemo z mešalnimi funkcijami. Mešalne funkcije so največkrat Bernsteinovi polinomi [18]. Bernsteinov polinom stopnje n, kjer

je i indeks kontrolne točke, je definiran kot:

$$B_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i}; 0 \leq t \leq 1 \tag{3.1}$$

Uporabljali smo Bezierove krivulje stopnje 3. Izpeljani Bernsteinovi polinomi za take krivulje so:

$$B_{0,3}(t) = (1-t)^3 \tag{3.2}$$

$$B_{1,3}(t) = 3t(1-t)^2 \tag{3.3}$$

$$B_{2,3}(t) = 3t^2(1-t) \tag{3.4}$$

$$B_{3,3}(t) = t^3 \tag{3.5}$$

Končna enačba krivulje p stopnje 3 pa je:

$$p(t) = \sum_{i=0}^{3} p_i B_{i,3}(t), 0 \leq t \leq 1 \tag{3.6}$$

Unity podpira risanje Bezierovih krivulj s pomočjo skript [31], za risanje in oblikovanje krivulj v urejevalniku scen pa trenutno nima podpore. Za to funkcionalnost potrebujemo vtičnik. Najdemo ga lahko v trgovini Unity Asset Store, drugje na Internetu[1] ali pa ga spišemo sami.

3.5 Raspberry Pi

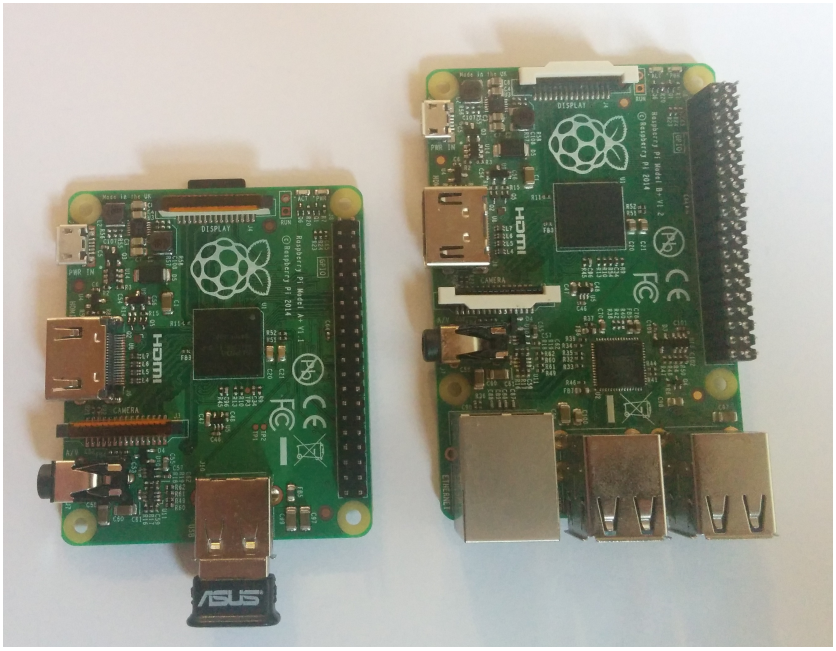
Za bolj izpopolnjeno realizacijo smo se odločili uporabiti napravo Raspberry Pi [26].

V osnovi je to majhen računalnik, ki ima poleg standardnih računalniških vhodov in izhodov (*HDMI*, *Ethernet*, *Bluetooth*, itd.) tudi nožice za priklop drugih elektronskih naprav. Na njem teče operacijski sistem Linux, in sicer posebna distribucija, imenovana *Raspbian*.

Na trgu je več različnih modelov. Za razvoj smo uporabljali model Raspberry Pi 1 B+, v končni različici izdelka pa zadostuje model A+, saj ima

[1] Po navadi v kakšnem Git repozitoriju.

manj vhodov in je zaradi tega manjši. Obstajajo tudi modeli 2 in 3 (v A in B variantah) ter bolj specializirani modeli, kot je npr. Raspberry Pi Zero.

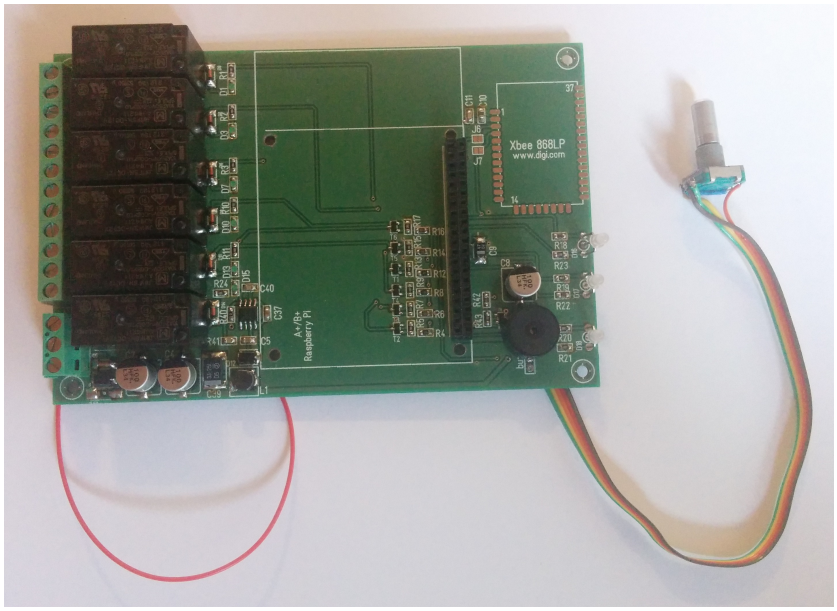


Slika 3.5: Raspberry Pi 1 A+ in B+

Slika 3.5 prikazuje uporabljeni napravi Raspberry Pi. Na levi je Raspberry Pi model A+, na desni pa model B+. Na sliki vidimo, da ima model B+ več vhodov USB, kar nam je olajšalo razvoj.

Operacijski sistem nam brez dodatne konfiguracije ponuja programiranje v jeziku C in Python. Odločili smo se za slednjega. Bistvene razlike v kodi ni, vendar imamo s Pythonom manj šablonske kode in hitrejši razvoj.

Ker so nožice na Raspberry Pi zelo zgoščene in podpirajo napetost samo do 5 voltov, so nam na voljo različne razširitvene plošče (angl. *extension boards*), na katerih so že zvezani različni uporabni elementi, kot so svetleče (LED) diode, gumbi, potenciometri in releji. Podobno razširitveno ploščo smo uporabili tudi mi.



Slika 3.6: Slika uporabljene razširitvene plošče

Slika 3.6 prikazuje uporabljeno razširitveno ploščo za Raspberry Pi. Nanjo je povezan element s funkcionalnostjo gumba in kodirnika (desno).

3.6 Android Studio

Pogon Unity3D sicer podpira razvoj aplikacij za platformo Android, vendar ne omogoča povezave Bluetooth prek telefona. Za uspešno povezavo Bluetooth smo zato morali napisati aplikacijo Android, ki deluje kot posredniška storitev med aplikacijo Unity in knjižnico Android Bluetooth. Za pisanje te storitvene aplikacije smo uporabili orodje Android Studio, uradni Googlov *IDE* za razvoj aplikacij Android.

Poglavje 4

Testiranje tablice Google Tango

Za testiranje tablice Tango smo uporabljali aplikacijo *Tango Explorer*,[1] ki smo jo omenjali že v poglavju 2.4.4. Aplikacija nam omogoča realnočasovni ogled oblaka točk, diagnostičnih podatkov, izris poti gibanja in pomnjenje ter razpoznavanje prostorov.

4.1 Zaznavanje globine

Zaznavanje globine lahko učinkovito spremljamo z opazovanjem oblaka točk. Testirali smo različne površine, oblike, razdalje in svetlosti okolja.

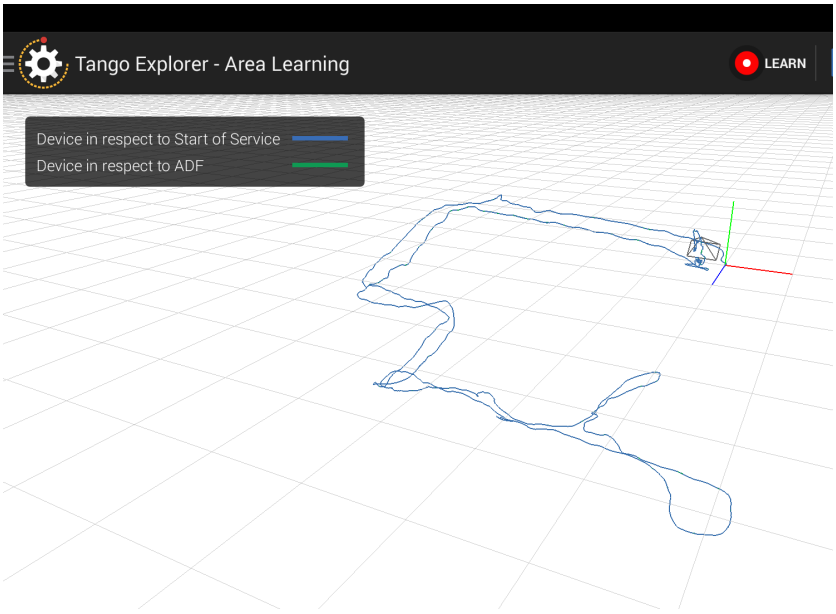
Zaznavanje je zelo dobro na ravnih površinah, tudi če smo nanje usmerjeni pod kotom (lahko smo skoraj paralelno). Zatakne pa se pri neravnih površinah. Če opazujemo valjast objekt, bo sprednji del objekta zaznal, vendar bodo ob straneh manjkale točke, saj je tam iz 2D-perspektive površina skoraj paralelna pogledu. Težavo predstavljajo tudi nepravilnosti majhnih objektov, npr. pri rastlinah — veliko drobnih vej in listov, ki jih Tango ne more zaznati in ločiti. Stekla in prozornih površin ne zaznava ("vidi" skozi njih). Google za snemanje priporoča razdaljo 0,5—4 metre od snemalne površine, kar je tudi meja zaznavanja globinskega senzorja. Na to nas

[1]Povezava do aplikacije: `https://apkpure.com/tango-explorer-tool/com.projecttango.tangoexplorer`

opozarja tudi aplikacija, če smo preblizu ali predaleč. Na objektih, ki padejo izven tega intervala razdalje, ne bo zaznanih točk. Svetlost in barva objektov tudi igrata vlogo. Tango zelo slabo zaznava črne površine. Bele površine sicer zazna, vendar lahko ob določenih pogojih odbijajo preveč svetlobe in s tem onemogočijo zaznavanje. Ob močni sončni svetlobi se lahko podobna situacija pojavi tudi pri površinah drugih barv. Zaznavanje onemogoči tudi pretemen prostor — za delovanje na splošno Tango potrebuje dovolj svetlobe. Testirali smo tudi zaznavanje globine v zunanjih prostorih, vendar smo takoj naleteli na omejitev oddaljenosti štirih metrov. V notranjih prostorih štirje metri niso dolga ali omejujoča razdalja, vendar s to omejitvijo snemanje zunanjih pokrajin na daleč ni mogoče.

4.2 Sledenje gibanja

Eden izmed prvih testov, ki smo jih izvedli, je obsegal gibanje po različnih prostorih v stanovanju in opazovanje Tangovega sledenja gibanja (brez pomnjenja prostorov). Tango dobro zaznava orientacijo tablice. Prav tako pa na videz tudi zelo dobro zaznava in sledi 3D-gibanju (tudi v višino). S ponavljajočo hojo po prostorih smo ugotovili, da je čez čas sled postajala rahlo zamaknjena. To pa je povsem normalno, saj so brez pomnjenja prostorov edine reference orientacija tablice, pospeški in slika kamere. Če so vse površine preblizu/predaleč ali pa je prostor pretemen, Tango izgubi prostorsko predstavo in trenutni položaj na poti "odplava". To lahko dosežemo tudi s sunkovitimi gibi ali tresenjem tablice. Ob vrnitvi v ustrezne pogoje za delovanje se bo sledenje gibanja nadaljevalo, vendar bo naša pot imela veliko napako. Težavo reši pomnjenje prostorov.



Slika 4.1: Prikaz sledenja gibanja skozi dve nadstropji

Slika 4.1 prikazuje sledenje gibanja skozi dve nadstropji in nekaj hodnikov. Začeli smo v stičišču rdeče, zelene in modre črte (na desni strani). Nato smo naredili krog — spustili smo se za dve nadstropji (na levi strani), zakrožili naokoli in se povzpeli nazaj do začetne točke. Vidimo, da smo končali relativno blizu začetku. Upoštevati pa je treba, da še tako majhna napaka vizualizacijo v našem primeru zaradi majhnih eksperimentov zelo zamakne (tudi do nekaj centimetrov).

4.3 Pomnjenje prostorov

Najpomembnejši Tangov koncept je pomnjenje in zaznavanje prostorov, saj je naš glavni cilj zaznavanje eksperimentov.

Prvi test je zajemal pomnjenje stanovanja. Vse, razen bolj temnih prostorov, je Tango odlično zaznaval. Nato smo si zapomnili posamezne prostore, ki jih je tudi odlično zaznaval. Ugotovili smo, da je dovolj stati na sredini prostora in temeljito posneti celoten prostor okoli sebe. Ni nam treba skrbeti

za vsako podrobnost, vendar če želimo bolj zanesljivo zaznavanje v kakšnem večjem delu ali kotu prostora, je bolje, da se med snemanjem prestavimo bližje. Tablice nam med zaznavanjem ni treba imeti v popolnoma enakem položaju in orientaciji, kot smo jo imeli pri snemanju. Po naših testiranjih je z malo truda dovolj, da na tablici kamera vidi polovico prostora iz približno enake perspektive, kot smo ga posneli.

Zanimalo nas je tudi, kako dobro Tango zaznava majhne objekte. Začeli smo z majhnimi škatlami. Škatle smo snemali v zaprtem prostoru na dovolj veliki, skoraj enobarvni podlagi z neizstopajočim vzorcem. Sončno svetlobo smo zaradi konsistentnosti zastrli in za razsvetlitev uporabili luč v prostoru.

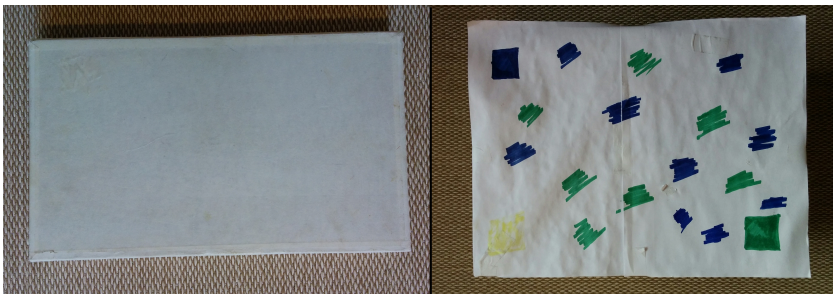
Majhno škatlo smo najprej posneli in si jo zapomnili, takoj zatem pa jo poskušali prepoznati. Tako kot zaznavanje prostorov, je tudi v tem primeru Tango hitro zaznal zapomnjeno "škatlo". Težava pa je nastala, če smo škatlo premaknili v drug prostor, z drugo podlago. Ugotovili smo, da si je Tango bolj zapomnil okolico (podlago) kot samo škatlo. Zaznavanje bo v takem primeru uspešno tudi na prvotni snemalni podlagi brez škatle. To je prva omejitev — objekt, ki si ga hočemo zapomniti, mora biti dovolj velik.

Naslednji cilj je bil uporabiti dovolj veliko škatlo, da bo lahko razdalja med snemanjem vsaj 0,5 metra, obenem pa bo v objektivu kamere večinoma samo škatla. Poskusili smo kar s škatlo električnih eksperimentov Elektro Pionir, ki smo jo kasneje uporabili za izdelavo prototipa (slika 5.2 v poglavju 5). Škatla meri 42 centimetrov v dolžino, 4 centimetre v višino in 25 centimetrov v širino. Dolga in široka je ravno dovolj, da na razdalji 0,5 metra v objektivu kamere prevladuje. Zaznavanje škatle na drugih podlagah je bilo ob podobnih svetlobnih pogojih in podobnih barvah podlag uspešno. Še vedno pa smo lahko brez škatle uspešno zaznali prvotno snemalno podlago.

Škatla oziroma pokrov kompleta Elektro Pionir je precej pisan, zato smo želeli testirati tudi enobarvno škatlo podobnih dimenzij. Zato smo uporabili kar spodnji del (dno) kompleta Elektro Pionir, ki je bele (zaradi starosti že malo sivkaste) barve. S tem smo zagotovili podobne oziroma skoraj enake dimenzije. Na tej škatli smo ponovili enak test. Zaznati nam jo je uspelo le na

snemalni podlagi, na drugih podlagah pa sploh ne. Tudi v tem primeru smo lahko uspešno zaznali snemalno površino brez škatle. Iz tega sledi ugotovitev, da je treba imeti čim bolj pisane škatle. S tem bo lahko Tango bolj natančno upošteval barvno informacijo snemanega objekta.

Poskusili smo še z nekaj kartonastimi škatlami, ki jih dobimo v trgovini. Za ta test smo nato uporabili kartonasto škatlo, ki meri 40 centimetrov v dolžino, 20 centimetrov v višino in 29 centimetrov v širino. Je malo krajša od škatle Elektro Pionir, vendar je precej višja. Uspešnost zaznavanja je bila le nekoliko boljša kot pri čisto beli škatli, kar lahko pripišemo višinski razliki. Zato smo upoštevali ugotovitev o pisanih škatlah in na (rjavo) kartonasto škatlo nalepili pisan papir. S tem smo zelo izboljšali zaznavanje in prišli na podobno raven, kot je pokrov Elektro Pionirja. Z uporabo višje škatle smo pridobili informacijo o obliki (višini), saj bomo v praksi eksperiment opazovali iz različnih smeri in pod različnimi koti. Dodatna informacija o višini v teh primerih izboljša zaznavanje in natančnost prikaza narisanih grafičnih elementov.



Slika 4.2: Slika prikazuje slab (levo) in dober (desno) primer podlage/škatle.

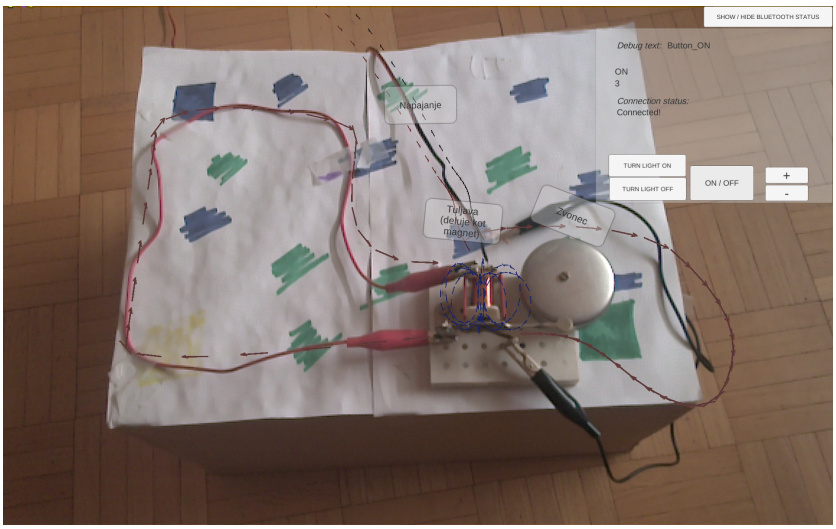
Na sliki 4.2 sta prikazani dve škatli. Škatla na levi je slab primer podlage, saj je površina skoraj povsem bela. Na desni strani pa je na sicer rjavo škatlo pritrjen pisan papir, ki ga Tango lažje zaznava in si jo tako bolje zapomni in razpozna. Poleg tega je tudi višja. To škatlo smo uporabili za izdelavo prototipa.

Poglavje 5

Implementacija

V tem poglavju podrobno predstavimo implementacijo aplikacije.

Cilj diplomskega dela je bila vizualizacija električnih tokov in magnetnih polj na primeru električnih eksperimentov z uporabo tablice, ki podpira platformo Google Tango. Za večjo interaktivnost smo eksperiment želeli tudi krmiliti preko tablice. Vizualizacijo smo dosegli z risanjem animiranih puščic, večjo interaktivnost pa z uporabo naprave Raspberry Pi in povezave Bluetooth.



Slika 5.1: Zaslonska slika aplikacije Tango

Slika 5.1 prikazuje zaslonsko sliko aplikacije, ki teče na tablici Tango. Na opazovan eksperiment (zvonec) nam dorisuje grafične elemente (puščice, besedilo), zgoraj desno pa prikazuje stanje povezave Bluetooth in kontrolne gumbe. Rdeče puščice prikazujejo tok, modre puščice pa magnetno polje, ki ga ustvari tuljava. Na zgornji žici je napajanje. Na njej tok iz pozitivnega pola vira napetosti rišemo z rdečimi, tok proti negativnemu polu vira napetosti pa s črnimi puščicami.

5.1 Pregled postavitve in aplikacije

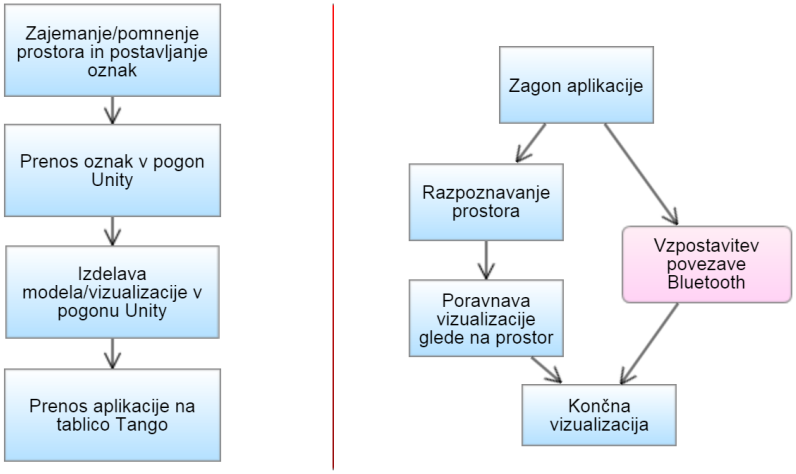
Za postavitev testnih eksperimentov smo uporabili star komplet *Elektro Pionir* (slika 5.2).

Slika 5.2: Komplet Elektro Pionir

To je komplet električnih elementov z navodili za postavitev različnih eksperimentov. Za našo nalogo so zadostovali podloga za pritrjevanje elementov, tuljava, tolkalo in zvonec. Dodatno smo potrebovali še nekaj žic in vir napetosti. Za slednje smo uporabili kar stare telefonske polnilce (napetost 5 voltov).

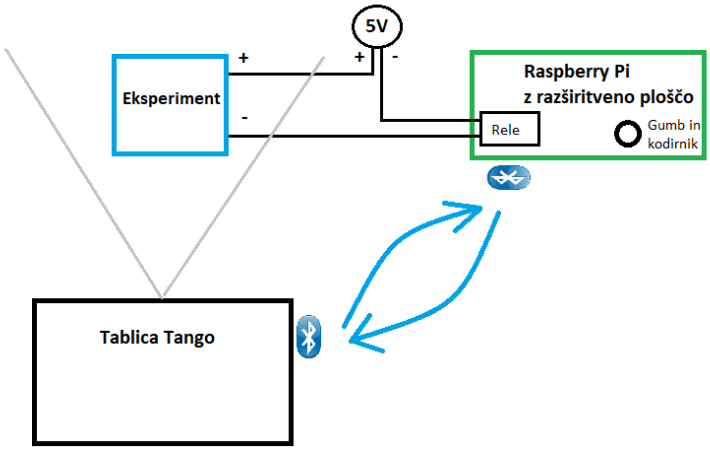
Eksperimenti iz kompleta Elektro Pionir so zelo osnovni, saj so namenjeni učenju pojavov elektrike in magnetizma. Zasnovani so tako, da jih vklapljamo in izklapljamo ročno s stikanjem žic. Zraven ni nobenega gumba ali "nadzorne plošče" za vklop/izklop in upravljanje delovanja.

To manjkajočo funkcionalnost smo pokrili z uporabo naprave Raspberry Pi. Preko tega majhnega računalnika smo hoteli krmiliti eksperimente, torej vklop/izklop in, glede na eksperiment tudi kakšne druge parametre (npr. hitrost pri elektromotorju). Obenem smo želeli fizično stanje eksperimenta imeti sinhronizirano s prikazom na tablici Tango. To smo dosegli z uporabo komunikacije Bluetooth med Raspberry Pi in tablico.



Slika 5.3: Shema razvojnega procesa in delovanja aplikacije

Slika 5.3 prikazuje razvojni proces (leva stran) po uspešno postavljenem eksperimentu in delovanje aplikacije (desna stran). Pri zagonu aplikacije sta obvezna razpoznavanje prostora in nato postavitev grafičnih elementov glede na prostor. Povezava Bluetooth ni obvezna, kar je označeno z drugačno barvo. Faze razvoja na levi strani so podrobneje razložene v nadaljnjih poglavjih, v nadaljevanju pa najprej opišemo še postavitev eksperimenta.



Slika 5.4: Prikaz povezav med napravami za splošni eksperiment

Slika 5.4 prikazuje povezave med napravami na primeru splošnega eksperimenta. V našem primeru je bil to zvonec. Plus vira napetosti je povezan na plus eksperimenta, minus pa na rele, ki je na razširitveni plošči Raspberry Pi. Drugi vhod releja je povezan z minusom eksperimenta. Z relejem krmili Raspberry Pi. Na razširitveno ploščo imamo povezan tudi element, ki združuje funkcionalnosti gumba in kodirnika. Ob pritisku na gumb Raspberry Pi vklaplja in izklaplja rele. S kodirnikom lahko tudi povečujemo in zmanjšujemo vrednost, ki se lahko uporabi kot neki parameter v eksperimentu (npr. hitrost elektromotorja). Svoje stanje (vklop/izklop ali spremembo vrednosti kodirnika) Raspberry Pi preko povezave Bluetooth sporoča tablici Tango. S tablico lahko opazujemo eksperiment, čigar stanje je zaradi komunikacije z Raspberry Pi sinhronizirano s prikazom. Preko tablice lahko Raspberry Piu tudi pošiljamo signale (vklop/izklop, sprememba vrednosti), tako da nam ni treba fizično pritiskati gumba ali vrteti kodirnika. Komunikacija med Raspberry Pi in tablico Tango je torej dvosmerna. Stanje povezave Bluetooth je tudi prikazano z diodo LED na razširitveni plošči(sveti rdeče ali zeleno).

5.2 Raspberry Pi

Nožice na Raspberry Pi se imenujejo splošnonamenski vhodi in izhodi — *GPIO* (angl. *General purpose input/output*). V Pythonu lahko z njimi upravljamo s pomočjo modula `RPi.GPIO`.[1] Posamezni nožici lahko določimo, ali je vhod ali izhod. Izhod nastavimo, kadar nožico želimo programsko vklapljati in izklapljati. Kadar pa želimo samo preverjati stanje naprave, ki je nanjo vezana (npr. gumb), jo nastavimo kot vhod. Raspberry Pi ima digitalen GPIO novejši podpirajo tudi *PWM*), kar pomeni, da je vsaka nožica lahko samo v dveh možnih stanjih — 0 ali 1. 5 voltov na nožici predstavlja visoko stanje (1), 0 voltov pa nizko stanje (0). Poznamo tudi izraza nizka in visoka fronta.

Z izhodi je lažje upravljati kot z vhodi — s pomočjo metode `GPIO.output()` izberemo nožico in določimo stanje, za katerega želimo, da ga ima.

Primer: `GPIO.output(1, True)` — nožici 1 nastavimo visoko stanje. Upravljanje releja in svetlečih diod je torej enostavno.

Več možnosti imamo pri upravljanju in branju vhodov. Nožicam, ki jim nastavimo vhod, moramo tudi definirati t. i. upor "pull-up" ali "pull-down". Ti upori nam varujejo in stabilizirajo vhodni signal. Določajo pa tudi, kakšno vrednost bomo prebrali, saj je vhod pri načinu pull-down ravno obraten od vhoda pri načinu pull-up. Primer: če imamo na vhodu gumb na načinu pull-down, bomo ob nepritisnjenem gumbu na vhodu prebrali 0, ob pritisnjenem pa 1. Če zamenjamo na način pull-up, pa bo ravno obratno — ob nepritisnjenem gumbu bomo prebrali 1, ob pritisnjenem pa 0.

V splošnem imamo tudi dva načina branja vhoda, in sicer:

- povpraševalni način (angl. *pooling*) in
- prekinitve.

[1] Uradna dokumentacija je na naslovu: `https://sourceforge.net/p/raspberry-gpio-python/wiki/Outputs/`, vendar je pomanjkljiva, zato priporočamo tudi ogled drugih virov (npr. `http://raspi.tv/category/raspberry-pi`).

Pri povpraševalnem načinu programsko (vsako iteracijo zanke) preverjamo stanje na vhodu. V tem primeru uporabimo metodo `GPIO.input(pin)`, ki ji podamo številko nožice. Metoda nam vrne 0 ali 1. Ko zaznamo spremembo na vhodu, se ustrezno odzovemo. Ta način pa ima veliko pomanjkljivosti. Celotno preverjanje imamo v glavni zanki, kar lahko hitro postane nepregledno. Poleg tega za vsako iteracijo zanke preverjamo vse vhode in tako zapravljamo procesorski čas. Obenem pa se lahko med zaznavo spremembe vhoda in z ustreznim odzivom za trenutek spremeni še kateri drugi vhod, ki ga, zaradi izvajanja druge kode (odziva na prvi vhod), zgrešimo.

Te pomanjkljivosti pa rešuje način s prekinitvami (angl. *interrupts*). Ob spremembi na vhodu se procesorju (strojno ali programsko) pošlje signal (prekinitev), da je prišlo do spremembe. Procesor nato začne izvajati ustrezen prekinitveno servisni program (angl. *ISR - Interrupt Service Routine*). V Pythonu se prekinitve programirajo z dogodki (angl. *events*). Vse, kar moramo storiti, je definirati funkcijo, ki naj se izvede ob dogodku, in nato zvezati želeni dogodek s to funkcijo. Dogodek je oblike "ko se na nožici 1 spremeni stanje iz nizke v visoko fronto, izvedi določeno funkcijo". V kodi: `GPIO.add_event_detect(1, GPIO.RISING, callback=funkcija, bouncetime=300)`. Ker ob spremembi vhoda napetost še nekaj časa niha, določimo še *bouncetime* v milisekundah. To določa čas, ko po zaznanem dogodku ne bomo zaznavali novega primera tega dogodka. S tem lahko tudi npr. omejimo prehitro pritiskanje gumba.

Opis komunikacije Bluetooth s tablico je opisan v poglavju 5.6.

5.3 Pomnjenje prostora in eksperimenta

Za večjo natančnost postavitve grafičnih elementov za obogateno resničnost smo uporabili funkcionalnost pomnjenja prostorov (angl. *Area Learning*).

Tablica Tango nam omogoča, da prostor posnamemo in v svoji aplikaciji vklopimo prepoznavanje posnetega prostora. S tem dosežemo postavitev grafičnih elementov glede na prostor. V paketu Tango za pogon Unity je

tudi že nekaj primerov aplikacij. Med njimi je aplikacija, ki nam omogoča snemanje prostora, sočasno pa lahko med snemanjem postavljamo oznake (angl. *markerje*) različnih barv. Z oznakami lahko označimo posamezne dele eksperimenta, kar nam kasneje pomaga v Unityju pri postavljanju različnih grafičnih elementov. Po končanem snemanju koordinate, orientacije in tipe postavljenih oznak zapišemo na "disk" oziroma interni spomin tablice v obliki XML. Poleg postavljanja oznak smo aplikaciji dodali možnost zajemanja in shranjevanja oblaka točk v datoteko XML.

Po končanem snemanju določimo ime datoteke ADF, kjer se shranijo lastnosti prostora. Zatem smo pripravljeni na uvoz oznak v Unity.

Pred začetkom snemanja je treba poskrbeti za ustrezen prostor in podlago eksperimenta.

V primeru, da je eksperiment na statičnem mestu in ga dlje časa ne bomo premikali, to ne predstavlja tako velike težave. Posnamemo lahko ves prostor, skupaj z eksperimentom.

Vendar, če si želimo zapomniti samo eksperiment, brez prostora, v katerem se nahaja, moramo poskrbeti za podlago in način snemanja. To pride v poštev, če želimo eksperiment prestavljati po različnih prostorih. Pri dosegu tega cilja smo naleteli na kar nekaj ovir. Naši eksperimenti so po velikosti relativno majhni, kamera pa ni dovolj natančna, da bi zaznala tako majhne podrobnosti. Zaradi majhnosti je tudi težko snemati — ne smemo posneti preveč okolice, ker si bo v takem primeru Tango zapomnil okolico in ne eksperimenta. Ti eksperimenti nimajo veliko ravnih ploskev, kar še dodatno poslabša zaznavanje.

Naštete težave smo zaobšli s kompromisom, da smo eksperimente pritrdili na podlago ali škatlo, ki smo jih vedno prenašali skupaj z eksperimentom. Tako si Tango ni zapomnil samega eksperimenta, ampak podlago, na kateri je eksperiment pritrjen. Ugotovitve glede ustreznih podlag so opisane v poglavju 4 (Testiranje tablice Google Tango). S to metodo izgubimo malo konsistentnosti pri prikazu grafičnih elementov, saj smo pri snemanju zelo omejeni, pod kakšnimi zornimi koti snemamo eksperiment. Med gledanjem

pa bomo stopali naprej, nazaj, malo vstran, mogoče še okoli po prostoru. Ker Tango med snemanjem okolice ni videl, bo nad grafičnimi elementi težje izvajal popravke (*drift correction*).

Paziti pa je treba tudi na svetlobo v prostoru. Eksperimenta ne priporočamo imeti pod oknom, saj bo zaznavanje nekonsistentno. Svetloba se čez dan močno spreminja in to vpliva na število zaznanih značilnosti v prostoru. Pri zaznavanju moramo paziti, da tablico usmerimo na eksperiment pod podobnimi zornimi koti, iz katerih smo tudi snemali.

5.4 Uvoz v Unity

Shranjene datoteke XML oznak je treba uvoziti v Unity.

Da bi si olajšali delo, smo implementirali aplikacijo in vtičnik za Unity, ki nam oznake uvozita samodejno. Temu je namenjena tudi posebna scena Unity. Uvoz je sestavljen iz dveh delov: prenosa datotek XML na računalnik in nato uvoza v Unity.

Prenos na računalnik nam olajša aplikacija, ki na tablici postavi strežnik, nanj pa se poveže vtičnik Unity, ki ga najdemo pod *Tango* → *Import data (network)*. Tako v Unity dobimo seznam vseh datotek XML z oznakami. Po izboru nam strežnik pošlje datoteko in jo postavi v korensko mapo projekta Unity. S tem smo si prihranili brskanje po datotečnem sistemu Androida in ročno kopiranje datotek. Izognili smo se tudi nadležnemu hrošču Android: novonastale datoteke preko povezave USB MTP ne vidimo, dokler Androida ne zaženemo ponovno [32].

Uvoz datotek XML v Unity opravimo z vtičnikom, ki je v *Tango* → *Import data (local)*. Najdemo pot do datotek in s klikom na gumb uvozimo. V obstoječo sceno se v objekt *Import* dodajo postavljene oznake in morebitni oblaki točk. Iz scene se izbrišejo vsi nepotrebni objekti (npr. Client, ki služi za povezavo na strežnik). Sceno shranimo pod drugim imenom, kar pusti originalno sceno za uvoz nedotaknjeno. Končna scena vključuje vse potrebne objekte za uporabo funkcionalnosti Tango, vklopi se tudi razpoznavanje pro-

stora, ki smo ga uvozili.

Po uvozu v sceni vidimo oznake, tako kot smo jih postavili med snemanjem. S pomočjo oznak pa lahko na ustrezna mesta postavljamo in oblikujemo grafične elemente.

5.5 Oblikovanje grafičnih elementov

Za grafične elemente lahko uporabimo vse, kar nam ponuja Unity — osnovne 3D-objekte, 2D-objekte, uporabniški vmesnik, luči in različne učinke. V našem primeru želimo primarno vizualizirati tokove s puščicami. To bi lahko dosegli tudi z risanjem črt v OpenGL znotraj Unityja ali pa z uporabo 2D-objektov. Odločili smo se za 3D-puščice, ki so predstavljene z objekti tipa GameObject.

Razred GameObject je v Unityju korenski razred, od katerega dedujejo vsi drugi bolj specializirani razredi. Vsak objekt tipa GameObject lahko vsebuje več drugih objektov (tipa GameObject) in tudi različne komponente (npr. skripte).

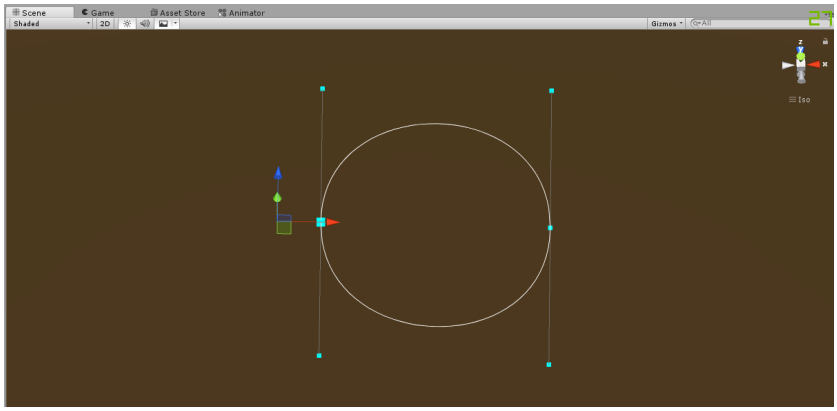
Ker smo želeli vizualizirati tok in magnetna polja, smo potrebovali bolj kompleksne in fleksibilne oblike krivulj. Zato smo so odločili za uporabo Bezierovih krivulj. Na internetu že obstajajo različne implementacije Bezierovih krivulj v Unityju. Izbrali smo implementacijo,[2] kjer je delovanje podrobno razloženo, omogoča pa tudi postavitev objektov na krivuljo [6].

Spodnja slika 5.5 prikazuje oblikovanje Bezierove krivulje v pogonu Unity.

Uporabljali smo kubične Bezierove krivulje. Takšne krivulje imajo dve točki, ki definirata začetek in konec, in dve kontrolnih točki, ki se nanašata na ukrivljenost pripadajočega dela krivulje. Ni nujno, da se krivulja začne in konča v isti točki.

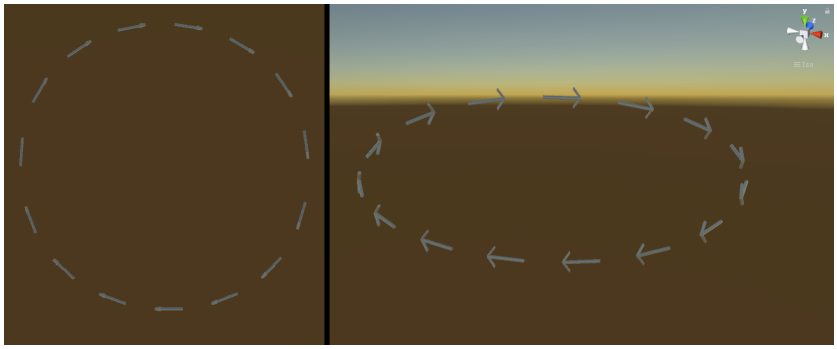
Ob zagonu aplikacije se krivulja ne izriše, ampak se po njej razporedijo objekti iz prej definiranega nabora objektov. To dosežemo z instanciranjem novih objektov razreda GameObject po celotni krivulji. V našem primeru

[2]Spletna stran z implementacijo: `www.catlikecoding.com`



Slika 5.5: Oblikovanje kroga z Bezierovimi krivuljami v pogonu Unity

je možen nabor objektov vseboval samo primerek ene puščice. Slika 5.6 prikazuje isti krog med delovanjem aplikacije iz dveh različnih zornih kotov.



Slika 5.6: Prikaz krivulje med delovanjem aplikacije iz dveh zornih kotov

Puščice so tudi animirane. Po krivulji se pomikajo od začetka do konca krivulje. Z dvema parametroma lahko nastavimo gostoto puščic na krivulji in hitrost premikanja. Krog na sliki in nekaj primerov drugih oblik krivulj je shranjenih kot *prefab*. To so v Unityju vnaprej shranjeni kompoziti objektov in komponent, ki služijo kot šablone. Z njimi prihranimo čas in poskrbimo za enotnost podobnih objektov v sceni.

Unity nam omogoča tudi gradnjo uporabniškega vmesnika. Vmesnik rišemo na objekt, imenovan platno (angl. *canvas*), ki vsebuje plošče (angl.

panels), gumbe, besedilo in druge gradnike. V naši aplikaciji poleg vmesnika, ki je statičen pogledu oziroma kameri, uporabljamo tudi vmesnik v svetu. Platnu smo definirali lastnost *worldspace canvas*, kar nam je omogočilo postavitev vmesnika v svet. To smo izrabili za različne opise in oznake eksperimentov. Na tak vmesnik lahko postavljamo tudi gumbe in druge kontrolne gradnike.

Ker lahko s tablico hodimo naokoli, smo za tak vmesnik morali poskrbeti, da je vedno obrnjen proti nam. To smo dosegli s skripto *LookAtCamera-Script.cs*, ki skrbi, da so vsi deli uporabniškega vmesnika vseskozi usmerjeni proti kameri.

5.6 Povezava Bluetooth in izdelava aplikacije

Prvotno smo hoteli povezavo preko Bluetootha realizirati z Unityem, vendar je podpora Bluetooth omejena na namizne platforme — preko povezave *serial*.

Povezavo smo morali implementirati s pomočjo posredniške storitve Android [5]. Storitev smo spisali v Javi z uporabo orodja Android Studio. Ob zagonu storitev poišče razpoložljive naprave, če najde Raspberry Pi, pa se nanj poskuša povezati. Pred tem smo Raspberry Pi in tablico morali spariti. Storitev vsa sporočila iz tablice posreduje Raspberry Piu, sporočila, ki jih prejme od Raspberry Pi, pa shrani v niz. Storitev smo napisali v celoti posredniško: skrbi samo za vzpostavitev povezave, vsa sporočila pa posreduje naprej.

Napisano storitev smo s pomočjo orodja Gradle [12] zapakirali v knjižnico s končnico .jar. To knjižnico smo postavili v projekt Unity, in sicer na pot *Assets/Plugins/Android/*. Zraven smo priložili še datoteko *AndroidManifest.xml*, ki vsebuje nastavitve za to storitev. Storitev se zažene samodejno ob zagonu aplikacije Unity.

V projektu Unity smo napisali skripto, ki preko te storitve komunicira z Raspberry Pi. S pomočjo razreda `AndroidJavaClass` smo se referencirali

na zagnano storitev. Objekt, ki nam ga je vrnil ta razred, nam je omogočil klicanje metod in branje ter spreminjanje spremenljivk v naši storitvi.

Na uporabniški vmesnik smo postavili nekaj oznak (angl. *label*) in gumbov. Oznake smo uporabili za izpisovanje stanja povezave in Raspberry Pi. Gumbi so bili namenjeni pošiljanju signalov preko povezave Bluetooth in ponovnemu poskusu ob morebitnem neuspehu.

Na strani Raspberry Pi smo za povezavo Bluetooth uporabili modul za Python, imenovan `python-bluez` [1]. Omogoča lažjo vzpostavitev povezave, branje in pošiljanje podatkov.

V prvi različici kode Python smo vhode GPIO brali še z uporabo povpraševalnega načina. To je zelo otežilo logiko komunikacije Bluetooth. Zaradi mešane logike (spremljanje vhodov GPIO, rokovanje s povezavo) sta bila posebno težavna dvosmerna komunikacija in zaznavanje prekinitve povezave. Zato smo prešli na način s prekinitvami (dogodki). Celotno logiko odzivanja na spremembe vhodov GPIO sedaj izvajamo v dogodkovnih funkcijah, v glavni zanki pa skrbimo izključno za komunikacijo s tablico. Na razširitveni plošči smo uporabili eno diodo LED za prikazovanje stanja povezave. Zelena barva pomeni delujočo povezavo, rdeča pa prekinjeno.

Obstoječo kodo smo dopolnili tako, da so dogodkovne funkcije spremembo stanja sporočale tablici.

Tablica se na sporočila odzove s spremembo prikaza stanja na uporabniškem vmesniku. Dodali smo še funkcionalnost, da se ob vklopu/izklopu eksperimenta grafični elementi pokažejo/skrijejo. Ker so se puščice ob zagonu prvotno generirale (instancirale) v korenu hierarhije objektov, smo ustvarili nov GameObject z imenom *Experiment*. Nastavili smo, da se puščice generirajo vanj, to pa nam je omogočilo izklop samo enega objekta za skrivanje vizualizacije celotnega eksperimenta.

Za večjo robustnost smo na Raspberry Pi napisali še nekajvrstično skripto *Bash*, ki inicializira Bluetooth. Nastavili smo, da se ta skripta in skripta Python zaženeta ob zagonu operacijskega sistema.

5.7 Težave

5.7.1 Dolgi razvojni cikli in razhroščevanje

V vseh fazah razvoja so nas spremljali dolgi razvojni cikli in razhroščevanje.

Ob testiranju aplikacije Tango smo morali aplikacijo preko Unityja naložiti na tablico, po končanem nalaganju iti do eksperimenta in preveriti delovanje. Med nalaganjem aplikacije na tablico je Unity neodziven, tako da spreminjanje scene med čakanjem ni bilo možno. Nerodno je bilo tudi razhroščevanje. Unity sicer ponuja metodo za testne izpise `Debug.Log()`, vendar za opazovanje izpisa potrebujemo povezavo USB in program *Logcat*, ki ga dobimo zraven Android Studia. Ker je kabel USB kratek, sta bila hoja okoli eksperimenta in sočasno branje izpisov nemogoča.

Podobno situacijo smo imeli pri povezavi Raspberry Pi - Android, saj smo za uspešno komunikacijo morali razvijtai na obeh sistemih hkrati.

Popravki storitve Android so tudi časovno potratni. V Android Studiu je treba spremeniti kodo, storitev zapakirati v .jar, prenesti v Unity in zgraditi aplikacijo Unity.

5.7.2 Nestabilnost Tango Core

Glavna aplikacija Tango oziroma storitev je Tango Core. Skozi razvoj in testiranje nam je ta storitev pogosto prenehala delovati. Mislili smo, da je težava v naših aplikacijah, vendar se ista težava naključno pojavlja tudi v uradnih Googlovih aplikacijah Tango.

Ob nepričakovani prekinitvi delovanja storitve Tango Core je treba tablico ponovno zagnati, kar lahko traja do ene minute.

5.7.3 Težave pri uvažanju v Unity

Prvotno smo hoteli uvoz realizirati tako, da tablico preko USB-ja priklopimo na računalnik in nato s pomočjo vtičnika uvozimo datoteke.

To pa z nobenim konstruktom Unity ni bilo mogoče. Tudi, ko smo do shrambe Android želeli dostopati s pomočjo dialoga za izbiro datotek, sistem Android ni bil viden. Poskušali smo tudi z bolj nizko-nivojskimi metodami v C#. IDE in Unity sta te metode poznala in prevedla, vendar nam je ob zagonu aplikacije javilo izjemo *Method not implemented*.

To je posledica uporabe C# Mono, ki je v primerjavi z glavno vejo razvoja C# na platformi .NET v zaostanku. Zato smo uvažanje razdelili na dva dela: prenos datotek XML s tablice na računalnik in ročni izbor datotek za dokončen uvoz v Unity. Prenos preko omrežja je tako nadomestil prenos na računalnik preko USB-ja.

Težava pa se je pojavila tudi pri uvažanju oblaka točk. Že z enim zajemom dobimo nekaj sto točk. Če to uvozimo v Unity, se bo pogon upočasnil do točke neuporabnosti.

5.7.4 Pozicioniranje elementov v Unityju

V osnovi je platforma Tango namenjena razpoznavanju prostorov in večjih objektov. V takšnih primerih se večjih napak in odstopanj ne opazi. V našem primeru pa je že nekaj milimetrsko odstopanje opazno.

Zaradi tega je potrebno zelo natančno pozicioniranje elementov v Unityu. Načeloma so nam v pomoč postavljene oznake, ki jih uvozimo, vendar se tudi med snemanjem te oznake ne postavijo povsem natančno. Če med snemanjem postavimo oznako predse na tla in se nato postavimo zraven nje, ne bo popolnoma na istem mestu (nekaj milimetrsko odstopanje). To se nato prenese v Unity. Težavo dodatno izostrijo eksperimenti z malo ali nič ravnimi ploskvami. Za postavitev oznake se namreč zaznava ravnina, na katero se nato z ustrezno orientacijo postavi oznaka.

Dodatno nam delo malenkost otežuje Unity, saj je mreža v urejevalniku definirana kot 1 enota = 1 meter. Ker delamo z enotami nekaj desetin ali stotin metra, nam kamera včasih nagaja.

5.7.5 Povezava Bluetooth

Težavi, ki smo ju imeli s povezavo Bluetooth, sta omenjana že pri drugih poglavjih, vendar jih bomo v tem poglavju opisali podrobneje.

Unity realiziranja povezave Unity Android - Bluetooth ne podpira na očiten način. Z jezikom C# lahko z razredom `Serial` dostopamo do priključenih vhodno/izhodnih naprav, vendar tak način v aplikaciji Android ne bi deloval. Na trgovini Unity Asset Store smo našli nekaj implementacij, ampak so bile vse plačljive.

Alternativa pa je bila izdelava posredniške *native* aplikacije(storitve) Android, ki ima neposreden dostop do knjižnic Android Java, vključno s funkcionalnostjo Bluetooth. Preko te storitve in knjižnic smo lahko upravljali z Bluetoothom, metode za pošiljanje, branje in vzpostavitev povezave pa smo izpostavili Unityju.

Na začetku nam je na Raspberry Pi uporaba povpraševalnega načina za preverjanje vhodov GPIO povzročala težave pri zaznavanju prekinitve povezave.

Da ne bi preveč bremenili procesorja, smo vsako iteracijo glavne zanke zakasnili za 10 milisekund, kar se pri preverjanju vhodov ni poznalo. Je pa to onemogočilo zaznavo prekinitve povezave in prožitev izjeme.

S prehodom na prekinitve smo se tej težavi izognili, saj se v glavni zanki z blokirajočo metodo samo čaka na podatke, druga logika pa je razporejena po dogodkovnih funkcijah.

5.7.6 Razširitvena plošča

Razširitvena plošča, ki smo jo uporabljali, ni bila uradna, ampak iz domače izdelave. Kot referenco smo dobili shemo plošče, vendar nas je kak teden pestila težava nedelujočih relejev in svetlečih diod.

S pomočjo izdelovalca plošče smo to težavo hitro rešili, smo pa izgubili nekaj časa z razhroščevanjem in dvomom delovanja Raspberry Pi ter našim razumevanjem električne sheme.

Poglavje 6

Rezultati

Google Tango je v osnovi zasnovan na prepoznavanju in gibanju po prostorih. Tudi Google je naprave Tango uporabil za vizualizacijo muzejskih eksponatov [11], kar obsega več prostorov, eksponati pa so relativno veliki.

Rezultati tega diplomskega dela pa kažejo na uporabnost naprav Tango pri vizualizaciji manjših objektov. Za konkretno področje smo si izbrali električne eksperimente majhnih dimenzij.

Na težavo smo naleteli takoj, ko smo poskušali posneti in si zapomniti samo eksperiment. Sam po sebi je majhen, poleg tega pa ima zelo malo ravnih ploskev, kar zmanjša verjetnost, da bo Tango našel dovolj lastnosti za konsistentno razpoznavanje. Ker na eksperimentu po navadi ni bilo dovolj lastnosti, si je Tango bolj zapomnil okolico (tla, podlago) kot eksperiment. Če smo eksperiment iz okolja odstranili in poskušali razpoznati okolico brez eksperimenta, je bilo prepoznavanje velikokrat uspešno, saj je Tango razpoznal okolico.

Po naših testiranjih Tango zelo konsistentno razpoznava okolico in prostore. Med snemanjem in razpoznavanjem ne smemo delati preveč sunkovitih gibov. Če med razpoznavanjem tablico stresemo, ji podremo prostorsko predstavo in sprožimo ponovno razpoznavanje.

Prišli smo do naslednjega zaključka: če imamo eksperiment na fiksnem mestu dalj časa, je priporočljivo posneti celoten prostor, skupaj z eksperi-

mentom. Ker se eksperiment ne bo premikal, se bo Tango referenciral na točke v prostoru in tako zagotovil konsistentno razpoznavanje in poravnano vizualizacijo grafičnih elementov. Če pa imamo eksperiment "mobilen", moramo poskrbeti za ustrezno podlago. Ker samega eksperimenta ne moremo konsistentno zaznavati, ga moramo pritrditi na podlago, ki jo bo Tango zaznal. Ta podlaga naj bo ravna, pisana in dovolj velika, tako da Tango med snemanjem ne bo zajel tal. Kot podlago lahko uporabimo tudi dovolj veliko škatlo, saj s tem zagotovimo še dodatno informacijo o globini in višini škatle. To je neke vrste kompromis med fiksnim in popolnoma mobilnim eksperimentom.

Na podlagi teh ugotovitev smo tudi implementirali prototip. Za podlago smo uporabili kartonasto škatlo, na vrh katere smo prilepili pisan papir. Nanjo smo pritrdili eksperiment in žice.

Ugotovili smo, da so grafični elementi vizualizirani relativno natančno. Ob hoji okoli eksperimenta se malenkost zamaknejo, vendar se ob vrnitvi v prvotno lego postavijo nazaj. Najbolj problematično je, če med snemanjem posnamemo preveč tal. Po navadi so na tleh ponavljajoči se vzorci. To lahko pripelje do tega, da Tango "razpozna" prostor z zamikom. V takem primeru so prekrivni elementi rahlo zamaknjeni. Ob opazovanju se bo Tango čez čas repozicioniral, lahko pa ga tudi stresemo in s tem "podremo" lokacijo v prostoru (*motion tracking*), kar bo povzročilo ponovno razpoznavanje.

Ker so eksperimenti majhni, bi si včasih vizualizacijo želeli pobližje ogledati. Če v takem primeru tablico preveč približamo eksperimentu, globinski senzor ne bo več zaznaval. Grafični elementi bodo v tem primeru "odplavali". Google za opazovanje priporoča razdaljo 0,5—4 metre.

Opazovanje grafičnih elementov od bližje pa razkrije tudi nenatančno pozicioniranje. V tem primeru ni kriv Tango, temveč postavitev elementov v Unityju. Zelo težko je namreč do milimetra natančno postaviti grafične elemente, predvsem na električne žice. Temu pa tablica tudi ni bila namenjena. Če opazujemo slike Googlove implementacije obogatene resničnosti za muzej, vidimo, da so vizualizirali velike ravne stene ali druge večje objekte, kot so

npr. sarkofagi. V teh primerih natančno pozicioniranje ni tako pomembno, je pa obenem lažje, saj je med snemanjem bolj enostavno postavljati oznake, pa tudi Tango boljše razpoznava večje objekte v prostorih.

Poglavje 7

Zaključek

Navidezna in obogatena resničnost sta ene izmed najbolj inovativnih tehnologij v računalništvu. Na trgu je tudi že veliko igralcev, ki hočejo to tehnologijo karseda izkoristiti. Eden izmed teh igralcev je Google s svojo platformo Tango za obogateno resničnost.

Ideja platforme Tango je, da jo lahko uporabljajo navadne mobilne naprave (mobiteli in tablice), ki izpolnjujejo minimalne strojne zahteve — vsebujejo vse zahtevane senzorje in imajo nasploh dovolj močno strojno opremo. Glavni koncepti so sledenje gibanju (angl. *Motion tracking*), pomnjenje prostorov (angl. *Area Learning*) in zaznavanje globine (angl. *Depth perception).* V medsebojnem sodelovanju nam te koncepti zagotavljajo prepoznavanje posnetih prostorov, pozicioniranje grafičnih elementov v teh prostorih in sprotno popravljanje napak.

Po naših testiranjih Tango prostore zelo dobro prepoznava, vendar ni primeren za zunanjo uporabo. Globinski senzor zaznava globino samo na razdalji približno 0,5—4 metre.

Cilj diplomskega dela je bil raziskati, kako dobro razpoznava manjše objekte, ki se po možnosti lahko prenašajo iz enega prostora v drugega. Za testiranje smo izbrali električne eksperimente manjših dimenzij.

Ugotovili smo, da za fiksen eksperiment, ki ga dlje časa ne bomo premikali, lahko posnamemo eksperiment in celoten prostor. Tako bo zaradi

več referenčnih točk zaznavanje boljše. Mobilen eksperiment pa sam po sebi Tango zelo slabo zaznava. Za konsistentnost zato priporočamo kompromis, da je eksperiment pritrjen na dovolj veliko, pisano podlago, ki jo lahko nato skupaj z eksperimentom prestavljamo. Paziti moramo, da v tem primeru ne posnamemo preveč okolice in da so prostori, v katere bomo prenašali eksperiment, približno enako svetli. S takim načinom izgubimo malo natančnosti razpoznavanja, vendar pridobimo mobilnost.

Z uporabo Unityja smo implementirali prototip. Za lažji razvoj smo razvili vtičnik, ki nam pomaga prenesti oznake, postavljene med snemanjem prostora, v Unity. Za oblikovanje in vizualizacijo grafičnih elementov (električni tok, magnetno polje) smo uporabili Bezierove krivulje. Za boljšo izkušnjo smo uporabili tudi Raspberry Pi, ki smo ga preko Bluetootha povezali s tablico. Tako smo lahko eksperiment vklapljali iz daljave s pomočjo tablice, prav tako pa smo dosegli sinhroni prikaz grafičnih elementov s fizičnim stanjem eksperimenta (vklopljen/izklopljen).

Pozicija grafičnih elementov je relativno natančna, vendar ob premikanju in pogledu iz različnih zornih kotov opazimo napake (nenatančnost).

Iz tega sledi sklep, da je Tango še vedno najprimernejši za razpoznavanje prostorov oziroma večjih objektov. Tudi izdelava grafičnih elementov je primernejša za večje objekte, saj ni potrebno tako natančno pozicioniranje.

7.1 Nadaljnje delo

Nadaljnje delo bi lahko usmerili še v druge funkcionalnosti, ki nam jih Tango ponuja.

Pred kratkim je iz eksperimentalne faze prišla funkcionalnost zaznavanja navidezne oznake (angl. *AR marker*), s katero bi lahko implementirali različne kretnje.

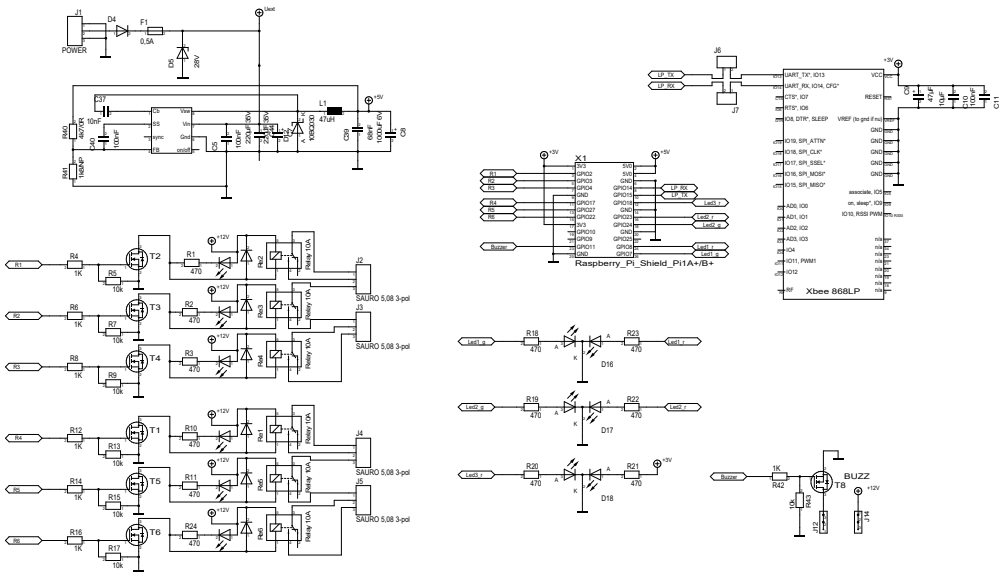
V eksperimentalni fazi je tudi grajenje objekta iz prostora (angl. *meshing*). S tem lahko prostor posnamemo in ga shranimo v format *obj*. Če bi to uporabili namesto ali pa v kombinaciji z oznakami pri snemanju prostorov,

bi bilo pozicioniranje elementov v Unityju lažje.

V primerih aplikacij Tango najdemo tudi primer povezave več naprav Tango v skupen svet. To bi lahko uporabili za prikaz iste vizualizacije vsem uporabnikom, dodali pa bi lahko kak zanimiv način medsebojne interakcije, komunikacije in sodelovanja.

Dodatek A

Shema razširitvene plošče



Slika prikazuje električno shemo uporabljene razširitvene plošče za Raspberry Pi. Sredinski element predstavlja Raspberry Pi in njegovo vezavo z drugimi elementi plošče. Levo spodaj je prikazana vezava relejev, levo zgoraj pa napajanje in na sredini spodaj svetleče diode. Poleg tega sta na desni še vezavi brenčala in modula za komunikacijo XBee, ki ju nismo potrebovali.

Literatura

[1] Android Linux / Raspberry Pi Bluetooth communication. Dosegljivo: http://blog.davidvassallo.me/2014/05/11/android-linux-raspberry-pi-bluetooth-communication/. [Dostopano: 13. 7. 2017].

[2] Area Learning — Tango — Google Developers. Dosegljivo: https://developers.google.com/tango/overview/area-learning. [Dostopano: 23. 5. 2017].

[3] Beziér curve. Dosegljivo: https://en.wikipedia.org/wiki/B%C3%A9zier_curve. [Dostopano: 13. 7. 2017].

[4] Oliver Bimber and Ramesh Raskar. *Spatial augmented reality: merging real and virtual worlds*. CRC press, 2005.

[5] Communication between an Android App and Unity. Dosegljivo: http://jeanmeyblum.weebly.com/scripts--tutorials/communication-between-an-android-app-and-unity. [Dostopano: 13. 7. 2017].

[6] Curves and Splines, a Unity C# tutorial. Dosegljivo: http://catlikecoding.com/unity/tutorials/curves-and-splines/. [Dostopano: 12. 7. 2017].

[7] Depth Perception — Tango — Google Developers. Dosegljivo: https://developers.google.com/tango/overview/depth-perception. [Dostopano: 23. 5. 2017].

[8] ECMA — Mono. Dosegljivo: http://www.mono-project.com/docs/about-mono/languages/ecma/. [Dostopano: 12. 8. 2017].

[9] Jan Egger, Markus Gall, Jürgen Wallner, Pedro Boechat, Alexander Hann, Xing Li, Xiaojun Chen, and Dieter Schmalstieg. HTC Vive MeVisLab integration via OpenVR for medical applications. *PLOS ONE*, 12(3):1–14, 2017.

[10] Five Augmented Reality Experiences That Bring Museum Exhibits to Life. Dosegljivo: http://www.smithsonianmag.com/travel/expanding-exhibits-augmented-reality-180963810/. [Dostopano: 12. 8. 2017].

[11] Google's AR platform Tango is going to let museum visitors explore exhibits. Dosegljivo: https://www.theverge.com/2017/1/9/14210956/google-tango-museum-ar-detroit. [Dostopano: 15. 7. 2017].

[12] Gradle Build Tool. Dosegljivo: https://gradle.org/. [Dostopano: 13. 7. 2017].

[13] Hunter G Hoffman, Walter J Meyer III, Maribel Ramirez, Linda Roberts, Eric J Seibel, Barbara Atzori, Sam R Sharar, and David R Patterson. Feasibility of articulated arm mounted Oculus Rift Virtual Reality goggles for adjunctive pain control during occupational therapy in pediatric burn patients. *Cyberpsychology, Behavior, and Social Networking*, 17(6):397–401, 2014.

[14] How the largest AR experience in the world was built. Dosegljivo: http://www.techworld.com/tutorial/developers/how-map-virtual-world-real-space-3661400/. [Dostopano: 12. 8. 2017].

[15] HTC Vive. Dosegljivo: https://www.vive.com/eu/. [Dostopano: 23. 5. 2017].

[16] Rabia Jafri, Rodrigo Louzada Campos, Syed Abid Ali, and Hamid R Arabnia. Utilizing the Google Project Tango tablet development kit

and the Unity engine for image and infrared data-based obstacle detection for the visually impaired. In *Proceedings of the 2016 International Conference on Health Informatics and Medical Systems (HIMS'15), Las Vegas, Nevada*, 2016.

[17] Juil Jeon, Myungin Ji, Juyoung Kim, Sangjoon Park, and Youngsu Cho. Design of positioning DB automatic update method using Google Tango tablet for image based localization system. In *Ubiquitous and Future Networks (ICUFN), 2016 Eighth International Conference on*, pages 644–646. IEEE, 2016.

[18] Matija Marolt. Parametrične predstavitve. Dosegljivo: `https://ucilnica.fri.uni-lj.si/pluginfile.php/50162/mod_resource/content/0/32%20Parametri%C4%8Dne%20predstavitve.pdf`, 2016. [Dostopano: 13. 7. 2017].

[19] Tomasz Mazuryk and Michael Gervautz. Virtual Reality History, Applications, Technology and Future. Technical Report TR-186-2-96-06, Institute of Computer Graphics and Algorithms, Vienna University of Technology, 1996.

[20] Microsoft HoloLens. Dosegljivo: `https://www.microsoft.com/en-us/hololens`. [Dostopano: 23. 5. 2017].

[21] Paul Milgram and Fumio Kishino. A taxonomy of mixed reality visual displays. *IEICE TRANSACTIONS on Information and Systems*, 77(12):1321–1329, 1994.

[22] Motion Tracking — Tango — Google Developers. Dosegljivo: `https://developers.google.com/tango/overview/motion-tracking`. [Dostopano: 23. 5. 2017].

[23] Alaeddin Nassani, Huidong Bai, Gun Lee, and Mark Billinghurst. Tag it!: AR annotation using wearable sensors. In *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, page 12. ACM, 2015.

[24] Oculus Rift. Dosegljivo: https://www.oculus.com/rift/. [Dostopano: 23. 5. 2017].

[25] Ramesh Raskar, Greg Welch, and Henry Fuchs. Spatially augmented reality. In *First IEEE Workshop on Augmented Reality (IWAR'98)*, pages 11–20, Los Alamitos, CA, 1998. IEEE Press.

[26] Raspberry Pi. Dosegljivo: https://www.raspberrypi.org/. [Dostopano: 23. 5. 2017].

[27] Tango. Dosegljivo: https://get.google.com/tango/. [Dostopano: 23. 5. 2017].

[28] Tango Developer Overview. Dosegljivo: https://developers.google.com/tango/developer-overview. [Dostopano: 23. 5. 2017].

[29] Carl Uddman Lindh and Johan Norberg. Augmented reality with holograms for combat management systems: Performance limitations for sonar tracks in a 3D map, presented with Microsoft HoloLens. Diplomska naloga, KTH, School of Information and Communication Technology (ICT), 2017.

[30] Unity - Game Engine. Dosegljivo: https://unity3d.com/. [Dostopano: 23. 5. 2017].

[31] Unity - Scripting API: Handle.DrawBezier. Dosegljivo: https://docs.unity3d.com/ScriptReference/Handles.DrawBezier.html. [Dostopano: 12. 8. 2017].

[32] Android MTP support does not show recent files until the device is rebooted. Dosegljivo: https://issuetracker.google.com/issues/36956498. [Dostopano: 12. 7. 2017].

[33] David Zeltzer. Autonomy, interaction, and presence. *Presence: Teleoperators & Virtual Environments*, 1(1):127–132, 1992. MIT Press.